THE UNIVERSITY OF ALBERTA

A LOW-COST MICROMETEOROLOGICAL DATA ACQUISITION SYSTEM

by

© PAUL EDWARD HOPPS

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

IN

METEOROLOGY

DEPARTMENT OF GEOGRAPHY

EDMONTON, ALBERTA

SPRING 1983

Abstract

The assembly of a portable, low-cost, microcomputer-based data acquisition system intended for the study of atmospheric turbulence is described. The system measures fluctuating temperature and wind with a platinum-wire thermometer, an inexpensive sonic anemometer and a pair of propeller anemometers. The performance of these sensors, with particular emphasis on frequency response, is examined. Analog conditioning, multiplexing, and 8-bit analog-to-digital circuitry, developed to interface the sensors to a low-cost RCA VIP-711 training microcomputer, are decribed. The microcomputer controls data collection at from 5 to 20 Hz, buffering up to 3328 bytes. Data collection is stopped while data blocks are transfered to standard audio cassette tape at 100 bytes per second. The microcomputer also retrieves the data from tape and transmits it to a host computer at 600 baud via an RS-232C interface. Documented assembly and machine language listings of all software are included. Suggestions are made for modifying the software to allow continuous data collection, variable sampling frequencies, faster data transfers to the host computer, and real time output.

The potential of the prototype system is demonstrated with examples of atmospheric spectra computed from data collected at a moderately uniform and horizontal site. Resulting power spectra and cospectra compare well with

results from others as well as with theoretical expectations in the inertial subrange. The effects of analog to digital resolution on the data are also examined. Eight bits are sufficient provided analog amplification and offset settings are properly used.

Acknowledgments

I would like to thank Dr. K. D. Hage for supervising this project. I would also like to thank Dr. J. Tartar and Dr. E. R. Reinelt for serving on my examining committee along with Dr. Hage.

The assistance of the following people during the field testing stage of this project is gratefully acknowledged: Karen Finstad, Ron Goodson, Peter Hof, Claude Labine, Laura Smith, and Leslie Stovel.

Funding for the instrumentation used in this project was supplied by the Natural Sciences and Engineering Research Council of Canada under Grant A1184.

Table of Contents

List of Symbols

C_p	specific heat of air at constant pressure
C_{ij}	cospectrum (i,j= u,v,w, or θ)
f	reduced wavenumber (=nz/U)
H	generalized amplitude function
H	vertical flux of sensible heat
L	Monin-Obukhov length
L	propeller anemometer length constant
L_o	axial propeller length constant
n	frequency (Hz)
n_o	Nyquist frequency
Δn	spectral frequency interval (Hz)
N	rate of dissipation of temperature variance, $\overline{\theta'^2}$
R	digitizing resolution
Ri	Richardson number
S_i	power spectra (i= u,v,w, or θ)
S_{ij}	cross spectra (i,j= u,v,w, or θ)
t	time
Δt	sampling period
T	generalized transfer function
T_f	transfer function of low-pass filter
T_p	transfer function of Gill propeller anemometer
T_{S1}	transfer function of a horizontal sonic anemometer
T_{S3}	transfer function of vertical sonic anemometer
u	longitudinal wind (x direction)
u_*	friction velocity

U	mean horizontal wind
v	lateral wind
w	vertical wind
x	coordinate aligned with the mean horizontal wind
z	vertical coordinate
α	Kolmogoroff constant for one-dimensional velocity spectra
β	Kolmogoroff constant for one-dimensional temperature spectra
ϵ	dissipation rate of turbulent kinetic energy
ζ	nondimensional stability parameter (=z/L)
θ	potential temperature
Θ	Angle-of-attack
κ	von Kármán constant
ρ	air density
σ^2	variance
$\sigma_n{}^2$	variance of digitizing noise
τ	surface shear stress
ϕ	phase function
ϕ_f	phase lag of low-pass filter
ϕ_p	phase lag of propeller anemometer

List of Tables

List of Figures

Figure | Page

Figure Page

Chapter 1

INTRODUCTION

Our understanding of the atmospheric surface layer has been greatly enlarged by the study of careful measurements of wind and temperature fluctuations over flat, uniform terrain. Parameters of particular interest include the vertical flux of heat,

$$H = \rho C_p \overline{w'\theta'}, \tag{1.1}$$

and the downwards momentum flux or surface shear stress,

$$\tau = -\rho \overline{u'w'} = \rho u_*^2, \tag{1.2}$$

where the overbars denote time averages and primes indicate fluctuations about a zero mean. In addition to time domain statistics such as these, much emphasis is now placed on the spectra of atmospheric turbulence. There is substantial experimental confirmation of the predicted inertial subrange behaviour in one-dimensional Eulerian spectra at frequencies above $n \simeq U/z$:

$$S_u(n) = \alpha (U/2\pi)^{2/3} \epsilon^{2/3} n^{-5/3}, \tag{1.3}$$

$$S_v(n) = S_w(n) = (4/3) S_u(n), \tag{1.4}$$

$$S_\theta(n) = \beta (U/2\pi)^{2/3} N \epsilon^{-1/3} n^{-5/3}. \tag{1.5}$$

The Kolmogoroff constants, α and β, are relatively well established, thus dissipation rates of energy, ϵ, and temperature variance, N, may be evaluated from measurements made at frequencies far below the actual dissipation range.

It is only in recent years that instruments capable of making reliable measurements of atmospheric turbulence at frequencies in the inertial subrange have become generally affordable. For the the all-important vertical wind measurements, the sonic anemometer is currently the instrument of choice. The basic principle of the device is simple; the difference in times for sound waves to travel in opposite directions along a fixed path is proportional to the wind speed along the path. However, the first commercial sonic anemometers, introduced in the late sixties, were complex and costly and therefore not widely used. Recently, following the work of Campbell and Unsworth (1979) and Larsen at al. (1979), single-component sonic anemometers for vertical wind measurements have become available for around $1000.[1] This is only a fraction of the cost of more established units and work by Shuttleworth et al. (1982) indicates further price reductions and performance improvements are still to come.

Flow distortions caused by the acoustic transducers of a sonic anemometer become a severe problem when horizontal wind measurements are attempted. Consequently, horizontal sonics seem likely to remain as very specialized (and expensive) research tools. The ionic anemometer, recently developed by Barat et al. (1982), may prove to be the low-cost answer to measuring horizontal wind fluctuations. This simple anemometer, based on the drift of ions created

----

[1]All prices are Canadian (1982)

during a corona discharge, is well suited to both vertical and horizontal turbulent wind measurements. At present, however, relatively low-cost, commercially-available instruments used to obtain horizontal wind variations include propeller anemometers (Hicks, 1972) and drag anemometers (Smith, 1980).

Much of the research aimed at developing low-cost turbulence instruments centers around an interest in obtaining reliable, routine (possibly even remote) flux measurements by use of (1.1) and (1.2). This so-called eddy correlation method has attracted much interest from workers in agriculture and forestry, as well as meteorology, who have traditionally used highly empirical, aerodynamic methods to estimate fluxes from profile measurements. This emphasis on flux measurements is reflected in the data acquisition products currently on the market. Still following an approach first used by Dyer et al. (1967), commercial units tend to be dedicated to computing and displaying real-time covariances. While analog circuitry was originally used for this purpose, modern systems are microprocessor-based and perform digital calculations. Either way, the drawback with such an approach is that it precludes any further analysis of the data. As a result, the study of atmospheric spectra continues to be done with on-site minicomputers (e.g. Redford et al., 1981) or multichannel FM analog recorders which are later digitized with a minicomputer-based facility. Both approaches are

expensive, cumbersome, and require a high degree of technical expertise.

For conventional meteorological applications, micro-processor-based data logging is becoming commonplace. The least expensive systems, which store data on audio cassette tape, start at about $2000. Retrieving and transmitting the data to a host computer for processing requires another device of comparable cost. Weihofen and Woehl (1981) have shown, however, that a commercial training microcomputer, costing less than $200, could be used in place of such a system. Their device saves data on audio cassette tape using the 300 baud (30 bytes per second) Kansas City Standard format. By using this popular audio encoding scheme, the tapes of Weihofen and Woehl could be read directly by the host computer system available for their data processing. Unfortunately, use of the Kansas City Standard results in a system that is too slow for collecting multiple channels of turbulence data.

The slow speed of the Kansas City format has led many microcomputer manufacturers to abandon it in favour of their own faster versions. Several training computers, in the class used by Weihofen and Woehl, are currently available with cassette interfaces operating at around 100 bytes per second which makes them suitable for collecting turbulence data. Such non-standard formats, however, are unlikely be supported by the host computer to be used for data processing. This means the microcomputer must be used to

control data retrieval as well as data collection.

The objective of this project was to assemble a microcomputer-based data acquisition system capable of collecting several hours of turbulence data and transferring these data to a host computer for processing. In addition to being inexpensive, it was desired that the system be easy to construct without the need for specialized development equipment. In order to accomplish this the system has been kept as simple as possible. The possibility of combining real-time calculations with the primary task of data collection, although desirable, has not been explored here.

Chapter 2

A LOW-COST INSTRUMENT ARRAY

2.1 Instrument Requirements

Sensors are the starting point of any data acquisition system. Those used to measure atmospheric turbulence require much better resolution and frequency response than are needed in conventional meteorological applications. As well, turbulence sensors must be carefully designed so that they do not significantly modify the characteristics of the flow which they are attempting to measure.

The response of an instrument to a sinusoidal input may be characterized by an amplitude function, $H(n)$, and a phase lag, $\phi(n)$, such that a sinusoidal input signal, $\sin(2\pi nt)$, is transformed to $H(n)\sin\{2\pi nt-\phi(n)\}$. Here, it is more convenient to use the transfer function

$$T(n)= H^2(n)= S'(n)/S(n), \tag{2.1}$$

where $S'(n)$ is the computed estimate of the actual power spectrum, $S(n)$. Knowledge of $T(n)$ reveals the limitations of an instrument and, up to a certain extent, enables them to be corrected. Typically, the half-power point, where $T(n)=1/2$, is taken as the representative cut-off frequency of a sensor. Figure 2.1, from McBean (1972), permits one to estimate the loss of vertical flux due to a high frequency cut-off as a function of height, wind speed, and stability. Typical conditions encountered at 5 to 10 m above the

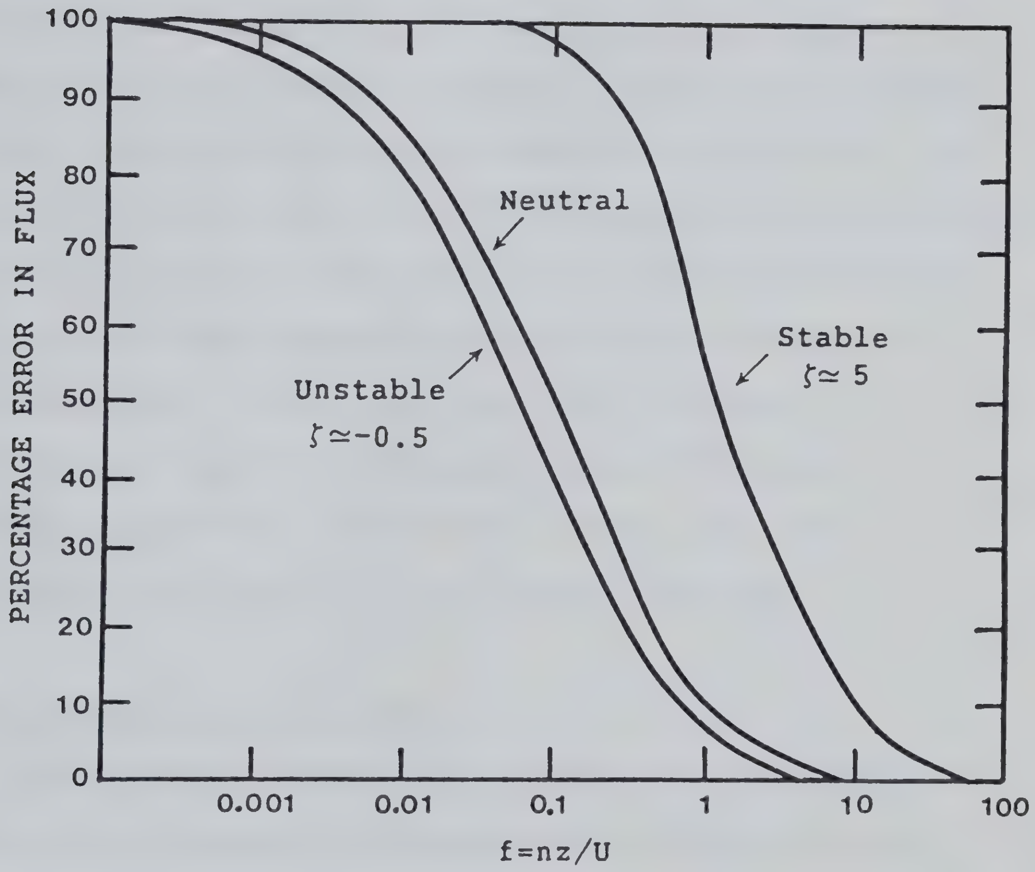


Figure 2.1. Approximate loss of vertical flux due to a sharp, high frequency cut-off (after McBean (1972)).

surface require cut-offs ranging from 1 to 5 Hz in order to keep the loss of flux under 10%.

Instrumental phase response is often ignored because power spectra are independent of it. Differences in $\phi(n)$ between two instruments will, however, effect covariances and cospectra. A complex cross-spectrum, S_{12}, may be corrected for instrumental response by subtracting $\Delta\phi=\phi_2-\phi_1$ from the phase and dividing the magnitude by by H_1H_2. The cospectrum may then be obtained from $Re\{S_{12}\}$ and numerically integrated to give an improved estimate of the covariance.

In the present project two propeller anemometers, a sonic anemometer, and a platinum wire thermometer are used to measure atmospheric turbulence. Each instrument represents a different trade-off between cost and performance which must be individually examined.

2.2 The Propeller Anemometer

The propeller anemometer, developed in the early sixties, is a durable, low-cost, fairly sensitive wind sensor. The widely-used Gill UVW array (consisting of three propeller anemometers mounted at right angles to each other) presently sells for around $1500.[2] Each anemometer consists of a lightweight, four-blade, helicoid propeller (23 cm diameter) directly driving a d.c. generator which produces approximately 50 mV/(m/s). Ideally, a fixed propeller anemometer would respond only to the component of the wind

----
[2]Made by R. M. Young Co., Traverse City, Michigan

vector parallel to its axis, thus allowing a fixed array of three orthogonal propellers to completely determine the wind field. In practice, Gill propeller anemometers deviate from this so-called cosine response as the angle of attack (angle between the rotation axis and the wind vector) increases. Drinkrow (1972) and Horst (1973a) have investigated this problem and devised iterative schemes for recovering the true wind components from those measured by an orthogonal array of propellers.

The frequency response of the Gill anemometer is of more serious concern. Typically, it is modelled with a transfer function of the form

$$T_p(n) = 1/\{1+(2\pi nL/U)^2\} \tag{2.2}$$

and a phase lag

$$\phi_p(n) = \tan^{-1}(2\pi nL/U), \tag{2.3}$$

where L is called the length constant. By examining the response of propellers released from rest in steady, wind tunnel flows, Hicks (1972) suggested that

$$L \simeq L_0 \cos^{-1/2}\Theta, \tag{2.4}$$

where L_0 is the length constant measured with the mean wind aligned with the anemometer (about 1.0 m for a 23 cm propeller) and Θ is the angle of attack. A half-power point at $n = U/2\pi L$ means that this sensor will frequently lose 20% or more of the vertical flux and rarely allow dissipation estimates from the inertial subrange without large

corrections to computed spectra. Figure 2.2 shows the large variations in T (n) due to changes in axial wind speed. In addition, individual variations due to bearing wear and dirt can also substantially degrade the frequency response. This makes applying meaningful corrections to propeller spectra difficult, since response lengths based on individual calibrations should be used, rather than those supplied by the manufacturer.

If one propeller is aligned directly with the mean wind ($\Theta=0°$) for optimum performance, (2.3) implies that the frequency response of an orthogonal propeller (lateral or vertical) will be extremely poor. Fortunately, Fichtl and Kumar (1974) have shown that the situation is not quite so bad in a turbulent atmosphere. For angles of attack near 90° they found that the response is primarily a function of the intensity of the turbulence along the propeller axis, resulting in length constants typically 4 or 5 times less than those given by (2.4). Even so, the frequency response is continuously varying and the transverse propeller is almost always operating in its non-linear regime below 1 m/s. Stalling occurs frequently as the axial wind speed falls below 20 cm/s.

For horizontal wind measurements, mounting a perpendicular pair of propellers such that the mean horizontal wind makes an angle of 45° with each, seems to be the best approach. In this special case, both propellers have the same response ($L \simeq 1.2$ m) and a single transfer function can

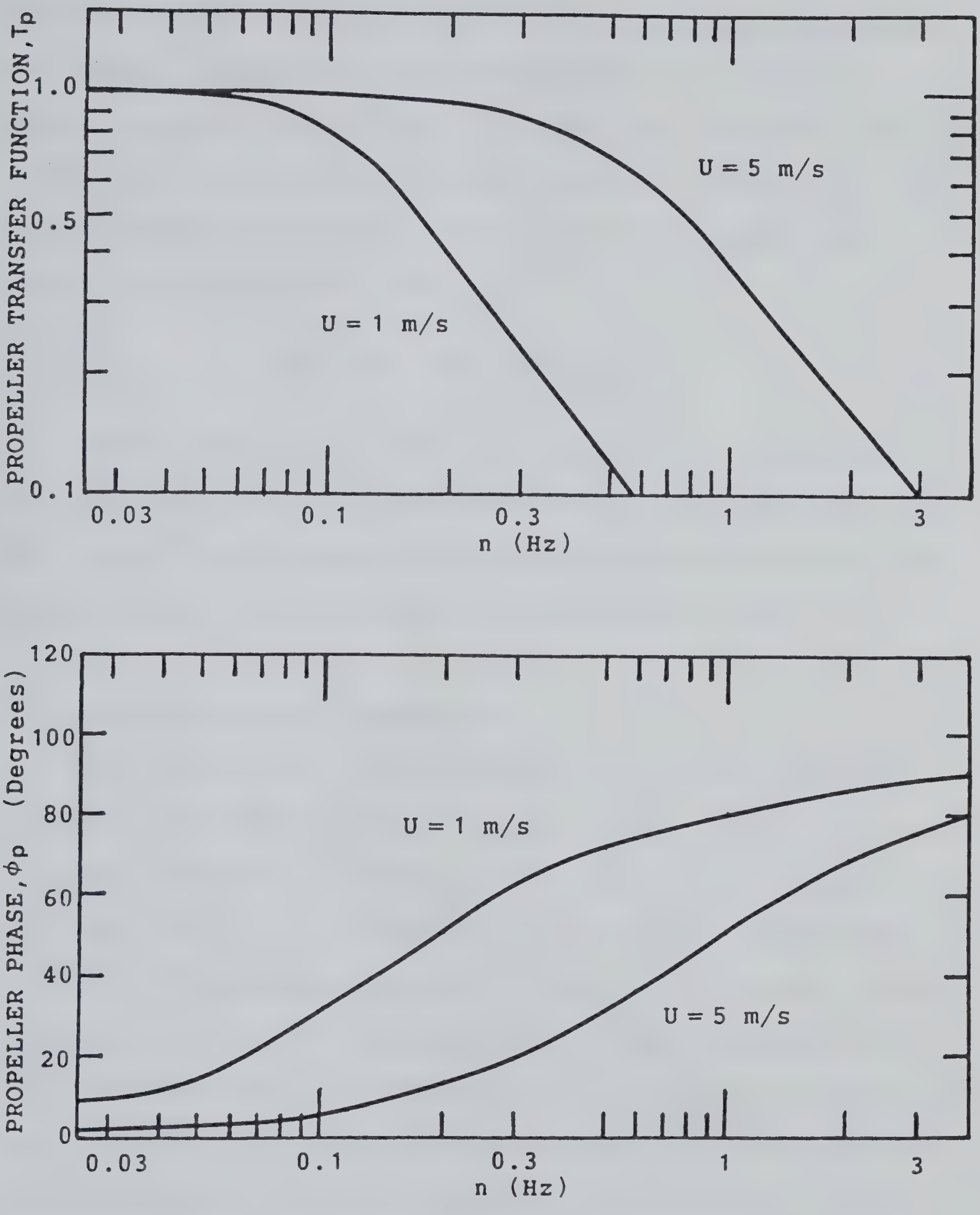


Figure 2.2. Transfer function (a) and phase lag (b) of a Gill propeller anemometer with L=1 m. The abscissa may also be read as reduced wavenumber, f, in which case the wind speed labels may be read as heights (i.e. 1 and 5 m).

be used to modify the spectra of the resolved longitudinal and lateral components. All other cases will involve combining data from sensors with different responses. Brooks (1977) derived an effective time constant, using (2.4), for a wind component resolved from a pair of perpendicular propeller anemometers, given by

$$L' = L\{1.093+.093\sin(4\Theta-\pi/2)\}. \tag{2.6}$$

Despite the limitations of the propeller anemometer, low-cost and durability have resulted in its widespread use. The lack of an inexpensive alternative has resulted in the use of a pair of horizontal Gill anemometers here.

2.3 A Low-Cost Sonic Anemometer

One-dimensional, sonic anemometers are well suited to vertical wind measurements because of their relatively good frequency response, linearity, and absence of a threshold. Flow distortions in the wakes of transducers, which greatly restrict the allowable angle of attack of horizontal sonic anemometers, are not a problem for vertically-mounted units since the mean wind is almost always transverse to the sound path. The CA27,[3] a commercial version of the continuous-wave sonic anemometer designed by Campbell and Unsworth (1979), was purchased for this project, at a cost of about $1200. The instrument is shown in Figure 2.3. Most previous sonics have used separate transmitters and receivers at each end of

----

[3]Made by Campbell Scientific, Logan, Utah

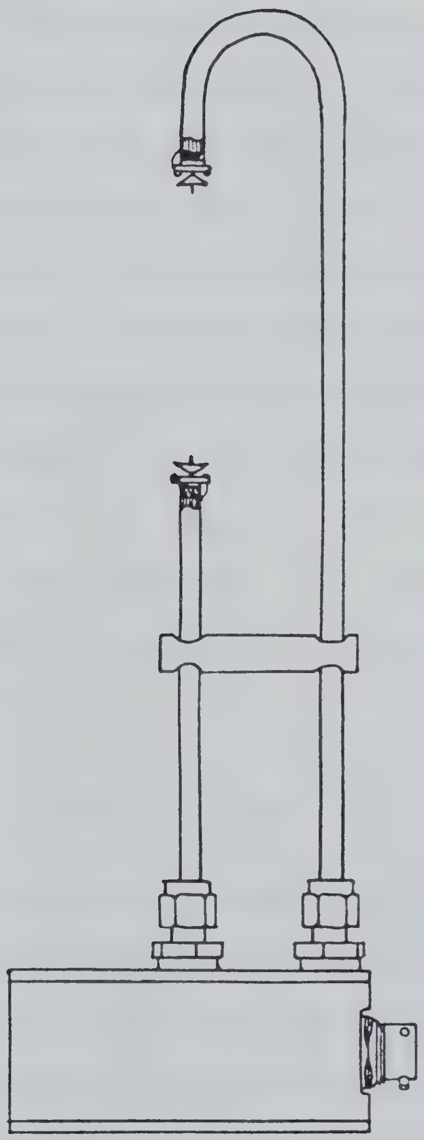


Figure 2.3. The CA27 Sonic Anemometer. Only a single ultrasonic transducer is used at each end of the 10 cm sound path.

the sound path. In the CA27, piezoelectric transducers interchange the tasks of transmitting and receiving ultrasound a hundred times per second. The unit operates from a 12 V battery, has an output of 1 V/(m/s) and a range of ±3 m/s. The effect of temperature variations on both the speed of sound and the characteristics of the transducers is not taken into account by the circuitry of this sonic anemometer, thus limiting accuracy to about ±1 cm/s.

Although susceptible to rain and dew, the small size of the transducers employed by the CA27 allow the use of a sound path only 10 cm long, half the distance used by most sonics. This is significant, as it is the averaging of the wind over this path which ultimately limits the high frequency response of the instrument. For the case of mean wind along a sound path of length d, Mitsuta (1961) derived the transfer function

$$T_{s1}(n)= \{(U/\pi nd)\sin(\pi nd/U)\}^2 \qquad (2.5)$$

by applying Taylor's frozen turbulence hypothesis, x=Ut, and integrating a sinusoidal wave over the time interval appropriate to the sound path. This approach also indicates that there is no phase shift associated with the horizontal line averaging process. A comparable analytic expression for a vertically mounted sonic anemometer has not been found, although Kaimal et al. (1968) and Horst (1973b) have produced numerical approximations based on the assumption of an inertial subrange. The following fifth-order polynomial

closely fits their tabulated data for a vertical sonic anemometer with a single, 10 cm sound path:[4]

$$T_{S3}(n) = \exp\{-0.0268-0.0664x-0.0662x^2-0.0201x^3 -0.000269x^4+0.0000383x^5\}, \qquad (2.6)$$

where $x=\ln(n/U)$. Plots of T_{S3} are shown in Figure 2.4. Although much better than that of a Gill propeller, the frequency response of the sonic anemometer still has a strong dependence on the mean wind. However, this correction is relatively small in the range of interest here and, since U is the only important meteorological variable involved, T_{S3} can be applied to spectral data with a high degree of confidence.

2.4 A Platinum Wire Thermometer

Fluctuations of temperature are much easier to measure than those of wind. Generally, thermocouples or resistance wires with small size and mass are used. A fine-wire thermocouple (Campbell Scientific 127), intended for use with the CA27 sonic anemometer, sells for over $300. Here, following Haugen et al. (1967), a single strand of 99.9999% pure platinum wire, 25 μm in diameter, was loosely strung parallel to the sonics path. This sensor, and accompanying circuitry, cost about $10.[5]

----

[4]The T_{S3} values (labeled T_3) in Table 3 of Kaimal et al. (1968) should each be displaced to the next-lowest tabulated frequency. The graph of T_3 in Figure 1 of that paper is, however, correct.

[5]Platinum wire was obtained from Alpha Co., New York. Cost (in early 1982) was $37 per meter.

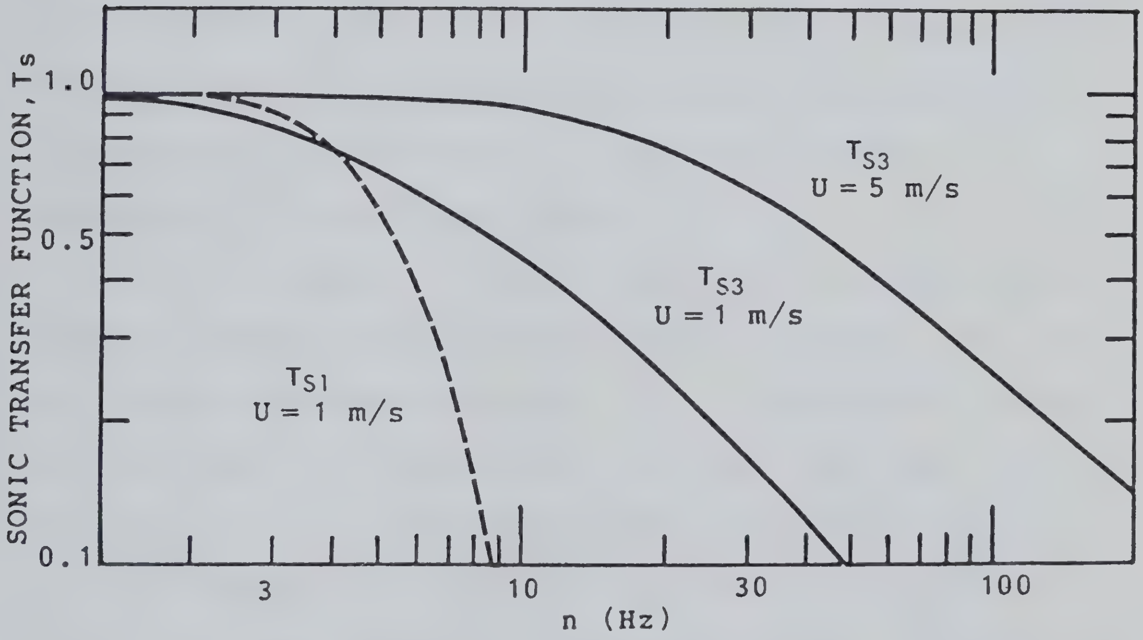


Figure 2.4. Transfer function of a vertical sonic anemometer with a 10 cm sound path. Also shown (dashed line) is the transfer function for a horizontal sonic with a 10 cm sound path. Below 5 Hz the two curves generally agree quite well.

The advantages of platinum are well known and include good stability and linearity over temperatures of meteorological interest. The change in resistance with temperature, 0.00395 R(0°C) per °C, is large compared that of most metals. The resistivity of pure platinum is 10.6 μohm-cm which, for the chosen diameter of 25 μm, gives a resistance of 2.16 ohm/cm. Kaimal (1968) has shown that the frequency response of such thin wire is predominantely determined by the line averaging process; therefore, the transfer function presented for the vertical sonic anemometer is directly applicable.

A 10 cm strand of the platinum wire (21.6 ohms) was placed in the Wheatstone bridge shown in Figure 2.5. A ten-turn, 10 ohm, variable resistor, R_a, is adjusted to bring the resistance of the platinum wire plus leads up to 28.0 ohms. This is easily done by nulling the bridge output, e_o. The fixed resistors (metal-film type with low temperature coefficient) were matched to better than 0.5%. Analysis of the circuit in Figure 2.5, under the assumption that ΔR, the variation of the platinum wire resistance, is much less than R, gives

$$e_o = E_1 \Delta R / 4R \tag{2.7}$$

In order to avoid self-heating and wind speed dependence, the current through the platinum wire must be kept low. Using E_1=5 V and R_1=1 kilohm gives e_1=135 mV and a current of 2.4 mA through the wire. Power dissipated by the wire is

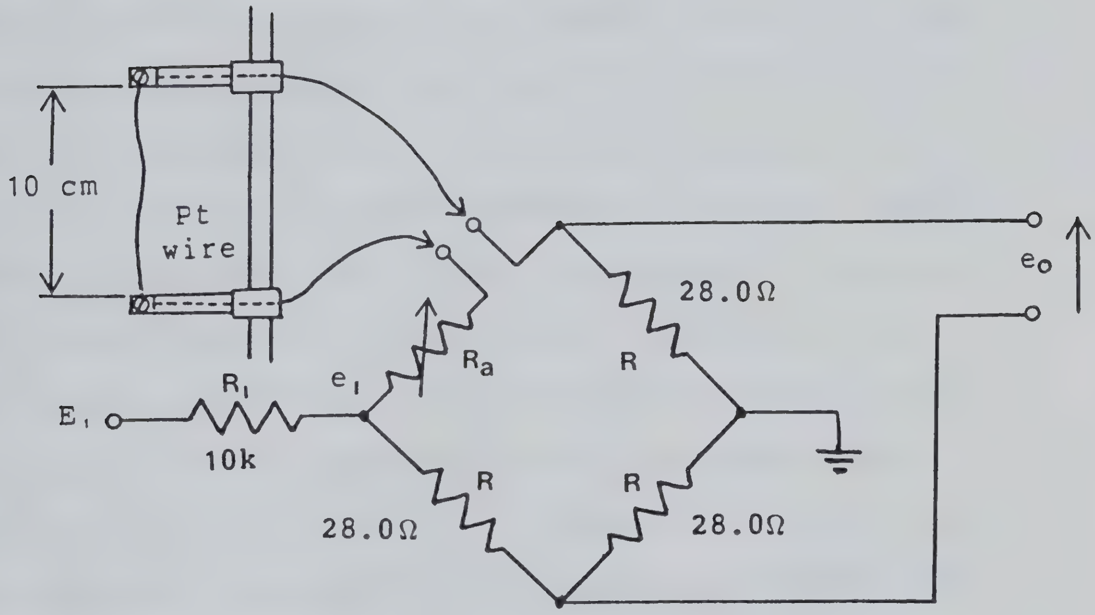


Figure 2.5. The platinum wire thermometer. The resistance of a 10 cm strand of 25 μm platinum wire is about 22 ohms. The temperature bridge adjust, R_a, is used to bring the active arm up to 28 ohms in order to balance the bridge.

approximately 150 μW. E_1 is provided by a 7805 voltage regulator which varies by less than 1% from 0°C to 30°C.

The bridge output, e_0, is very small, and must be applied to a high-gain, low-noise, differential amplifier. Using a gain of 5000, a 22 ohm strand of platinum wire was calibrated in a water bath which gives more stable and easily controlled temperatures than air. Using the range 20 to 35°C, a calibration coefficient of 0.107±0.004 mV/°C was obtained. The theoretical value for a 22 ohm wire is 0.105 mV/°C. From this good agreement it appears that a simple calculation based on the resistance of the platinum wire is sufficient for calibration. The calibration coefficient, in mV/°C, is obtained by multiplying the resistance of the platinum wire at room temperature (in ohms) by 0.00476. Figure 2.6, a comparison between the platinum wire thermometer and a commercial thermocouple (Campbell Scientific 127), clearly shows a high visual correlation. The effect of line averaging can be seen in the slightly smoother platinum wire trace.

2.5 Instrument Mounting

Mounting the selected instruments in the field is not a trivial task. The continuing controversy over the results of the Kansas experiment (e.g. Wyngaard et al., 1982) underscores the importance of avoiding obstructions which may distort the flow being measured. Proper alignment of the wind sensors is also vitally important. Wieringa (1972)

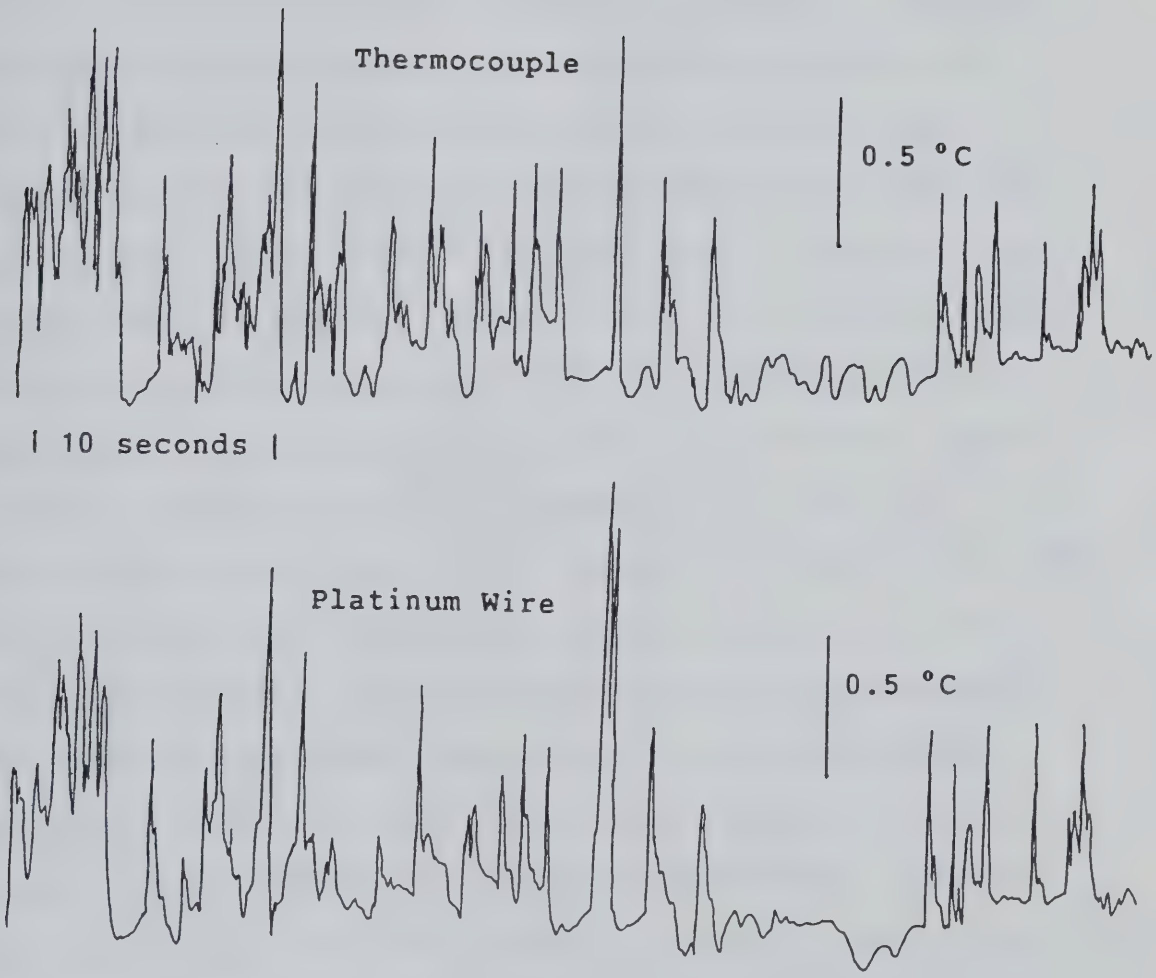


Figure 2.6. Comparison between platinum wire thermometer and a commercial, fine-wire thermocouple. The traces were obtained in the laboratory by gently blowing across the probes. The thermocouple was located about 2 mm from the center of the resistance wire. A high visual correlation is evident although the thermocouple seems to show more fine detail.

found that one degree of sensor tilt could result in errors of up to 10% in momentum flux and 5% in heat flux.

The CA27 sonic anemometer seems to be poorly designed in terms of minimizing obstruction to the flow. Hogstrom (1982) has recently found errors of 10 to 20% in variances and covariances caused by flow distortion around a vane-mounted hot-wire turbulence sensor which is much more aerodynamically shaped that the sonic anemometer used here. It therefore seems likely that distortion of the flow due to the cylindrical base of the CA27 could significantly effect the vertical wind measurement. To reduce this problem, the sonic probe was remounted on a seven pin, MS-type, Amphenol connector compatible with the standard Gill UVW array. The filter and blower housed in the base of the Gill array (used only for continuous operation applications with AC power) were replaced with the electronics of the sonic anemometer. The platinum wire probe was mounted on the sonic probe parallel to the sound path. The resulting wind-temperature array, shown in Figure 2.7, reduces obstruction problems while providing a single unit which is fairly easy to handle and level. Leveling by spirit levels and visual inspection can typically be done to within ±1°. When only a single level of measurement is attempted, the array should be mounted as high as possible in order to optimize sensor response.

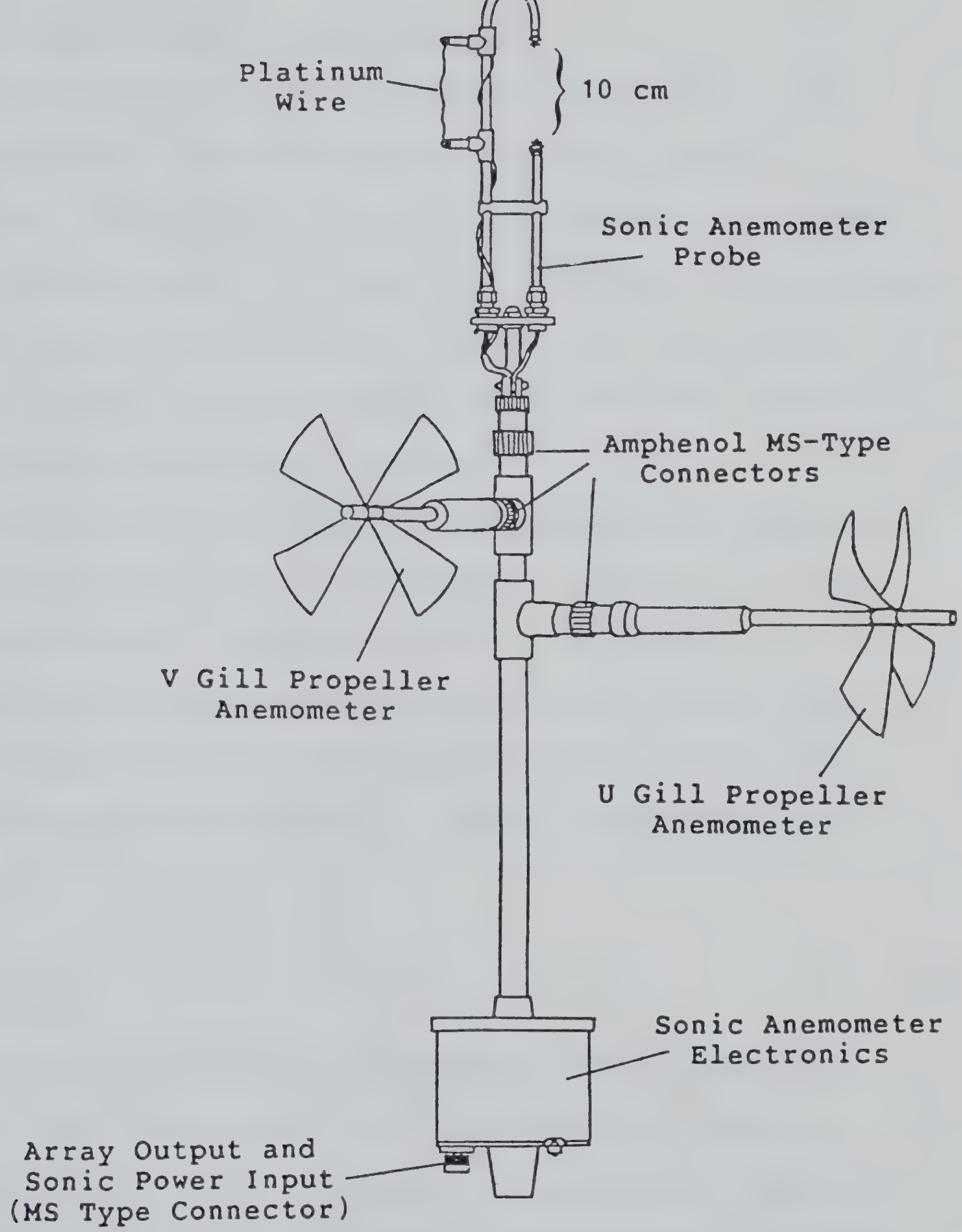

Platinum Wire
10 cm
Sonic Anemometer Probe
Amphenol MS-Type Connectors
V Gill Propeller Anemometer
U Gill Propeller Anemometer
Sonic Anemometer Electronics
Array Output and Sonic Power Input (MS Type Connector)

Figure 2.7. The wind-temperature array. The sonic anemometer and platinum wire thermometer mount into the socket on the Gill UVW array originally intended for a vertical propeller.

Chapter 3

ANALOG SIGNAL CONDITIONING

3.1 The role of Signal Conditioning

The outputs from the instruments, eventually to be digitized, must first undergo considerable analog processing. Typically, the small fluctuations of interest will be superimposed on a large d.c. component representing the mean value of the signal. This must be removed and the fluctuations must then be amplified so that they utilize the available digitizing range (nominally 0 to 5 V for all channels) as fully as possible. High-pass filters can be used to remove d.c. and low-frequency components of the signals, although, a simple offset, if possible, is preferred as it allows easy reconstruction of the absolute signal level. Finally, it is important that the signals undergo high-pass filtering in order to reduce errors due to aliasing.

3.2 Aliasing

Aliasing errors are introduced by the digitizing process. Only frequencies less than half the sampling frequency, the so called Nyquist frequency, $n_o = 1/2\Delta t$, can be resolved. Energy from the higher frequencies is folded back around the Nyquist frequency according to

$$S_a(n) = S(n) + \sum_{m=1}^{\infty} S(2mn_o - n) + S(2mn_o + n), \qquad (3.1)$$

where S_a the aliased estimate of the true power spectrum, S. Applying a rectangular filter prior to digitizing, which removes all frequencies above n_o, would eliminate aliasing completely, but such ideal filters are hard to come by. A practical, two-pole, low-pass, Butterworth filter with half-power point at n_o has a transfer function given by

$$T_f(n) = 1/\{1+(n/n_o)\}^4 \qquad (3.2)$$

and phase

$$\phi_f(n) = \tan^{-1}\{n_o n/(n_o^2-n^2)\}. \qquad (3.3)$$

These are shown in Figure 3.1. The phase lag of the filter can be ignored only if all of the channels experience the same filtering.

Given the natural $n^{-5/3}$ roll-off in the inertial subrange of most atmospheric spectra, a two-pole filter, although unable to completely eliminate aliasing, will confine the contamination to a single fold (negligible contribution from above $2n_o$).

$$S_f(n) = T_f(n)S(n)+T_f(2n_o-n)S(2n_o-n). \qquad (3.4)$$

As can be seen in Figure 3.2, $S_f(n)$ is a good approximation to the true spectra due to the close cancellation of aliasing by the two-pole filter. To further improve the spectral estimate it is necessary to provide an explicit expression for $S(2n_o-n)$. This can be done by extrapolating the measured spectra (not including the aliased end) or, following Kaimal et al. (1972), inertial subrange behaviour

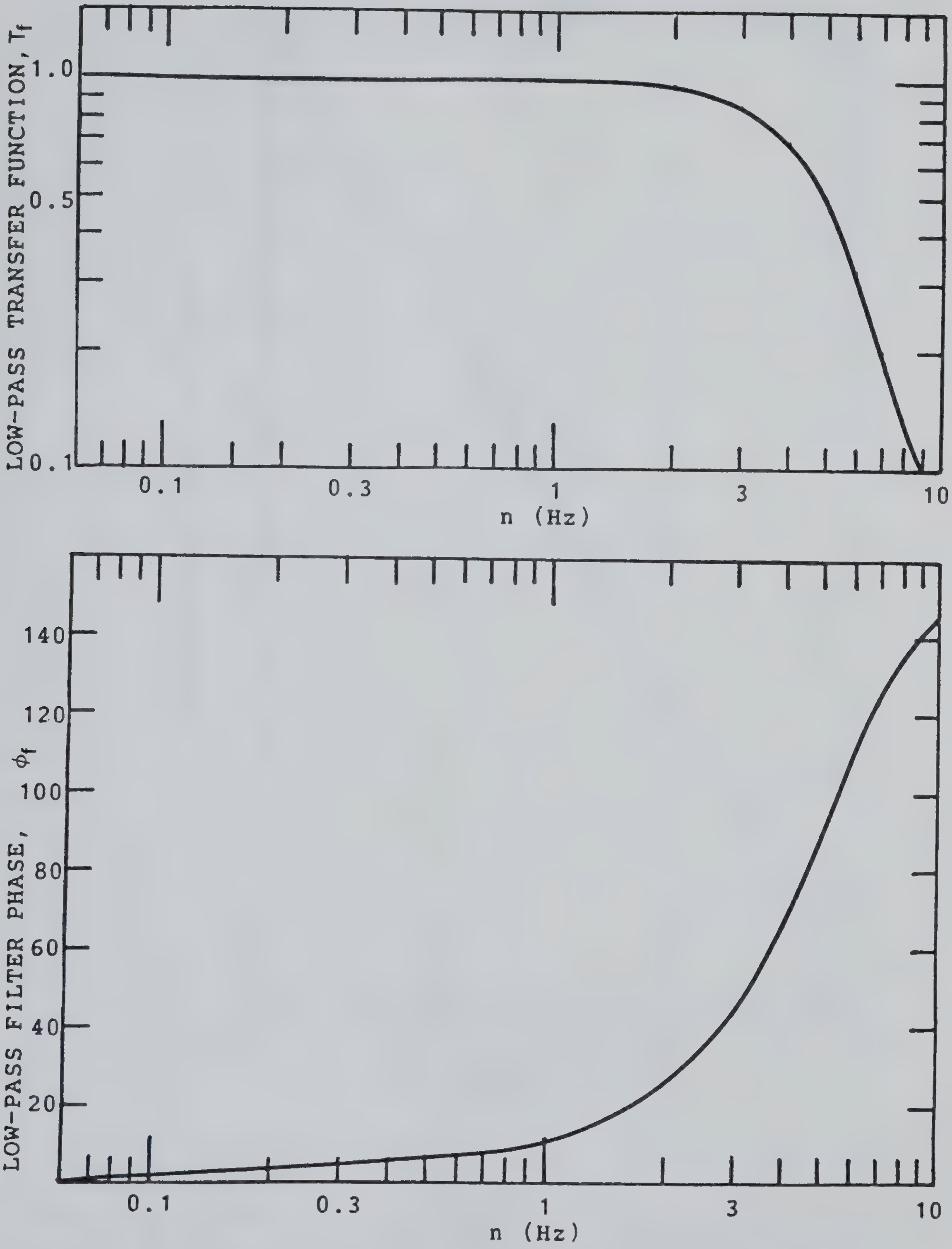


Figure 3.1. Transfer function (a) and phase (b) of a two-pole, low-pass, Butterworth filter. The half-power point is at 5 Hz.

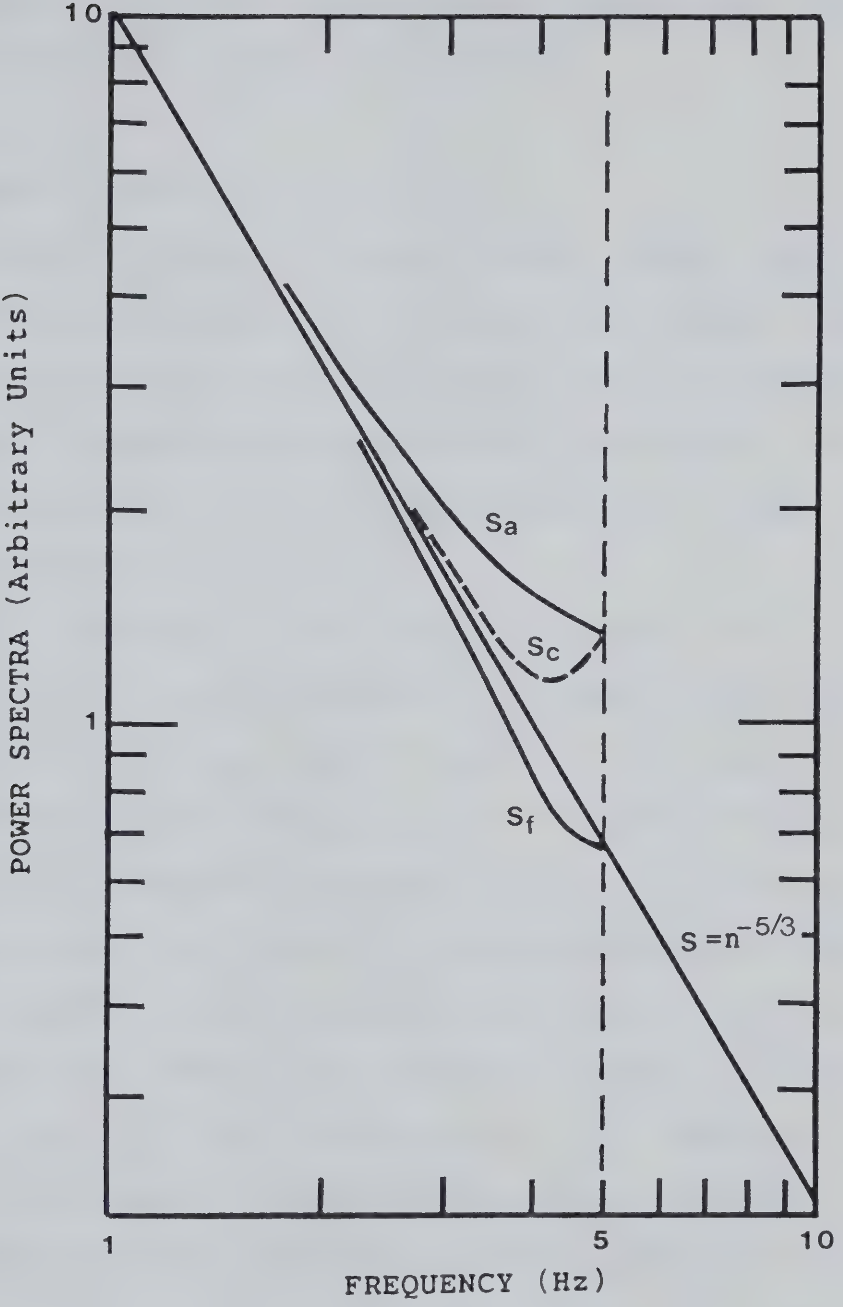


Figure 3.2. Effect of a two-pole, low-pass filter on aliasing in the inertial subrange. S_a represents the aliased estimate of S in the absence of any filtering. S_f shows the result of low-pass filtering with half-power point at the Nyquist frequency (5 Hz). Finally, S_c shows the result of correcting the filtered spectra, S_f, for the effects of low-pass filtering. While S_c is clearly an improvement over S_a, the best overall estimate appears to be S_f.

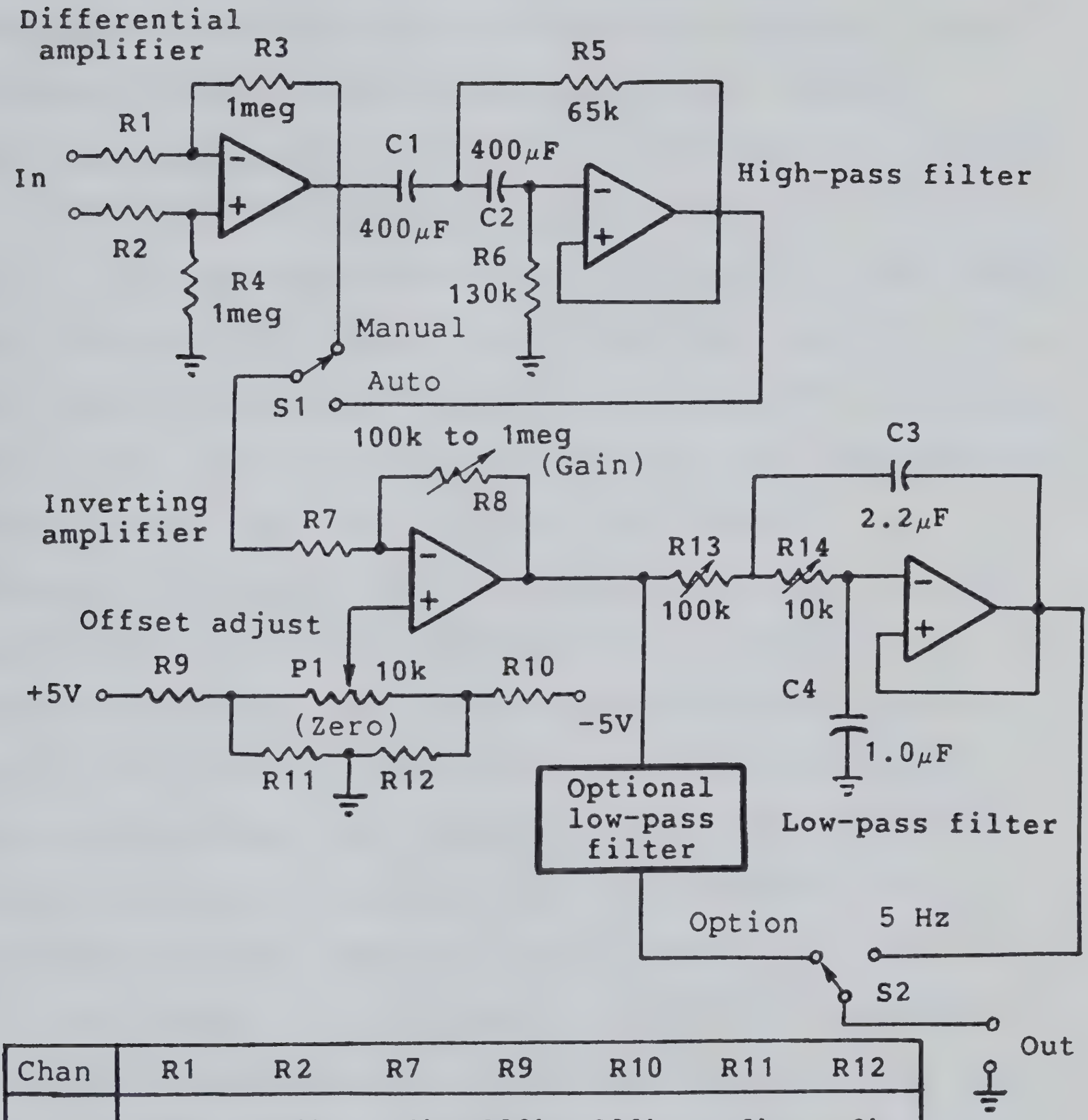
could be assumed. In most cases, however, this extra work will not appreciably improve the spectra.

3.3 Practical Circuitry

To provide the required amplification, offset, and filtering,the output from each instrument was passed through the precondition circuit shown in Figure 3.3. It consists of a differential preamplifier followed by a high-pass filter, an inverting amplifier with adjustable gain and offset, and a low-pass filter.

The major function of the differential amplifier is to reference all the incoming signals to a common ground. By reversing the role of the input terminals, this stage can also be used to set the final polarity. For most sensors $R_1= R_2= R_3= R_4=$ 1 megohm, thus giving unit gain and input impedance of about 500 kilohm (all resistors were matched to better than 0.5%). For the temperature signal, R_1 and R_2 were replaced with 100 kilohm resistors in order to give this stage a gain of 100. The resulting input impedance of only about 50 kilohms is not a problem, due to the low output impedance of the temperature bridge (less than 30 ohms).

The output from the differential amplifier then goes into a two-pole, Butterworth, high-pass filter with a half-power point at 0.005 Hz. The output from this filter connects into the inverting amplifier stage via the toggle switch S_1. This allows the high-pass filter to be left out



Chan	R1	R2	R7	R9	R10	R11	R12
T	100k	100k	2k	200k	200k	2k	2k
W	1meg	1meg	200k	10k	10k	10k	10k
U	1meg	1meg	20k	10k	10k	10k	10k
V	1meg	1meg	20k	20k	20k	5k	5k
X	1meg	1meg	20k	10k	10k	10k	10k
Y	1meg	1meg	20k	10k	10k	8.5k	8.5k

Spare channels

Figure 3.3. The prototype analog precondition circuit. Component values which vary from channel to channel are shown in the table. All resistors are 1% metal-film types. The operational amplifiers are each 1/4 of a TL084CN.

of the circuit if desired, in which case the output from the differential amplifier is sent directly to the inverting amplifier. The switch does not close off the input to the high-pass filter as this would require the filter to resettle (approximately 200 seconds) whenever S_1 is thrown.

The gain of the inverting amplifier stage is given by R_8/R_7 where R_8, constructed from a 10-position rotary switch and 100-kilohm resistors (1%), is variable from 100 kilohm to 1 megohm in 100-kilohm steps. The value of R_7 depends on the sensor. For the sonic anemometer R_7= 200 kilohm has been found to work well, giving gains from 0.5 to 5. For propeller anemometers R_7= 20 kilohm results in acceptable gains from 5 to 50. The platinum wire thermometer amplifier uses R_7= 2 kilohm, which, in conjuction with the gain of 100 in the differential amplifier, gives gains from 5000 to 50,000. The ranges were selected, on the basis of field tests, so that the typical range of turbulence levels can be accommodated with reasonable sensitivity.

The adjustable potentiometer arrangement attached to the noninverting terminal of the inverting amplifier stage allows a d.c. offset to be added to the signal, in order to approximately center it in the 0 to 5 V digitizing range. A ten-turn potentiometer was used for P_1 to allow good control over a wide range.

Finally, the signal is subjected to low-pass filtering. A two-pole, Butterworth filter with half-power point at 5 Hz is appropriate for the typical sampling frequency of 10 Hz.

However, to accommodate different sampling rates, the switch S_2 provides a path to an optional, interchangeable filter.

In the prototype system, the circuit shown in Figure 3.3, excluding the optional filter, was implemented on standard, one-sided, 22-pin, printed-circuit cards, each using a single, inexpensive, monolithic quad operational amplifier (TL084CN). Temperature dependent offset drifts are not critical in this application as the analysis of data invariably requires the removal of mean values and linear trends over a few minutes. The cost of parts, per channel, was about $15 although this could vary substantially depending on the quality of the hardware selected. Commercial quality integrated circuits, which have a recommended operating range of 0°C to 70°C, were generally used here. The prototype system includes two spare channels for a total of six. Optional filters for all six channels were implemented on a single printed-circuit card. The operational amplifiers in the system are powered by ±8 V obtained from a pair of 12 V lantern batteries via common monolithic voltage regulators (LM317,LM337).

3.4 Field Operation

The six channel prototype Analog Precondition Unit (APU) is shown in Figure 3.5. Two 12 V lantern batteries, which supply the power to the unit and the sonic anemometer, fit inside the unit. The basic wind-temperature array plugs into the rear panel of the APU with a single connector.

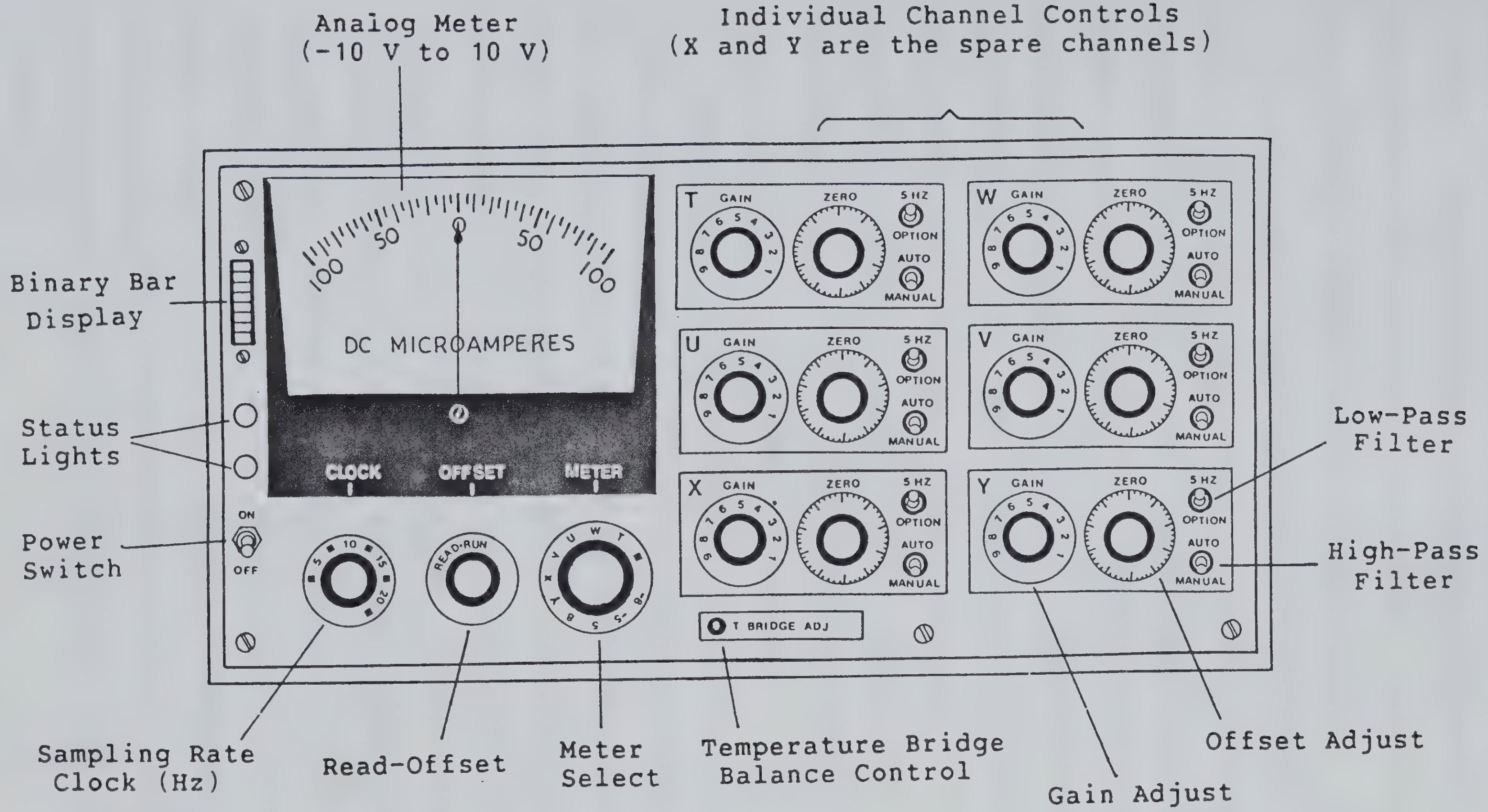


Figure 3.4. Front view of the prototype Analog Precondition Unit (APU). In addition to an analog precondition circuit board for each channel, this unit also houses the temperature bridge, power supply board, optional filter board, voltage reference, sampling clock, analog-to-digital board, and two 12 V batteries. The clock and A-to-D circuitry are discussed in Chapter 4.

Separate input connectors are provided for the two spare channels, X and Y. The channel switch allows any of the conditioned signals to be displayed on the analog meter. The offset switch allows reading the applied d.c. offsets.

In the field, it is necessary to adjust the gain and offset settings of each channel to obtain signals with means of about 2.5 V (25 μA on the analog meter) and fluctuations between 0 and 5 V (0 and 50 μA). The analog meter can be used for this purpose, however, a strip-chart recorder works much better. The recorder should be set on a 0 to 5 V range and connected to the meter output at the rear of the unit. If additional strip-chart recorders are available, individual channels can be monitored via conditioned signal outputs provided at the rear of the APU.

During field testing, it was found that high-pass filtering was never required for vertical wind measurements. For temperature, a fixed offset could generally be used for intervals of only about 15 minutes. The best gain settings at the start of an experiment were such that the maximum fluctuations covered from 50 to 75% of the available range. A slightly larger gain could be used if high-pass filtering was employed; however, filtering was avoided whenever possible due to the limitations it imposes on the low frequency contributions to fluxes and spectra. In addition, high-pass filter corrections are somewhat uncertain due to the difficulty of trimming and testing such low frequency filters.

Chapter 4

DIGITAL DATA COLLECTION

4.1 A Low-Cost Microcomputer

Following the most inexpensive approach possible, the conditioned analog signals were channeled through a multiplexer (MUX) into a single analog-to-digital converter (ADC). In order to operate correctly, the MUX and the ADC have to be provided with various control signals. The digital data produced by the ADC must then be collected, encoded as audio signals, and recorded on a cassette tape. These tasks may be performed easily and economically with the aid of a commercial microcomputer. The approach used in the present system is to collect the data temporarily in a reserved memory area in the computer. When this buffer is filled the collection process is stopped and the data are transferred to a cassette recorder. Data collection resumes when the tape transfer is complete and the cycle continues for as long as required. This is clearly not the best approach; continuous data collection would be preferable. However, the approach adopted is the simplest one possible and, therefore, was considered an appropriate starting point.

Numerous low-cost, single-board microcomputers, intended primarily for educational and recreational use, were available in early 1982 for under $300. The RCA COSMAC VIP CDP18S711, shown in Figure 4.1, was chosen for this

project although, unfortunately, it has since been discontinued by RCA. It is based on the 8-bit, COSMAC 1802 microprocessor running at 1.7609 MHz. The CMOS (Complementary Metal Oxide Semiconductor) fabrication of the 1802 and other components in the VIP-711 provide low power requirements and high noise immunity. On-board memory consists of two kilobytes of static RAM (Random Access Memory), empty sockets for an additional two kilobytes of RAM, and 512 bytes of ROM (Read Only Memory) containing an operating system. The VIP-711 also includes a cassette interface, video interface (an RF modulator is needed to use a TV without a video input), parallel I/O port, expansion connector, hexadecimal keypad, tone circuit, and a power supply. Except for the power supply, the system fits onto a single board measuring about 20 by 30 cm.

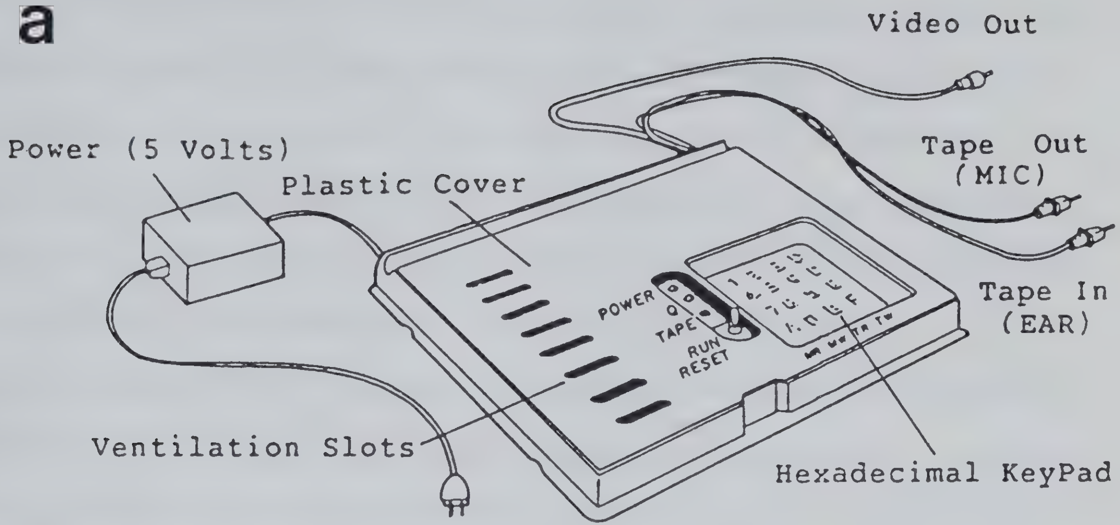
The basic VIP-711 cost $260. Adding a cassette tape recorder,[6] an RF modulator,[7] and filling the empty RAM sockets brought the price of the useable system to about $400. Various accessories which fit into the expansion connector were also purchased including a four kilobyte RAM expansion board ($150), an EPROM (Eraseable Programable Read Only Memory) programmer ($150), and EPROM interface board, ($60). These relatively expensive options were not needed for the current project.

The VIP-711's built in cassette interface, implemented largely in the operating system software, is vital to the

----

[6]Sony TCM-131, $80.
[7]M & R Enterprises Sup "R" Mod-II, $40.

a



b

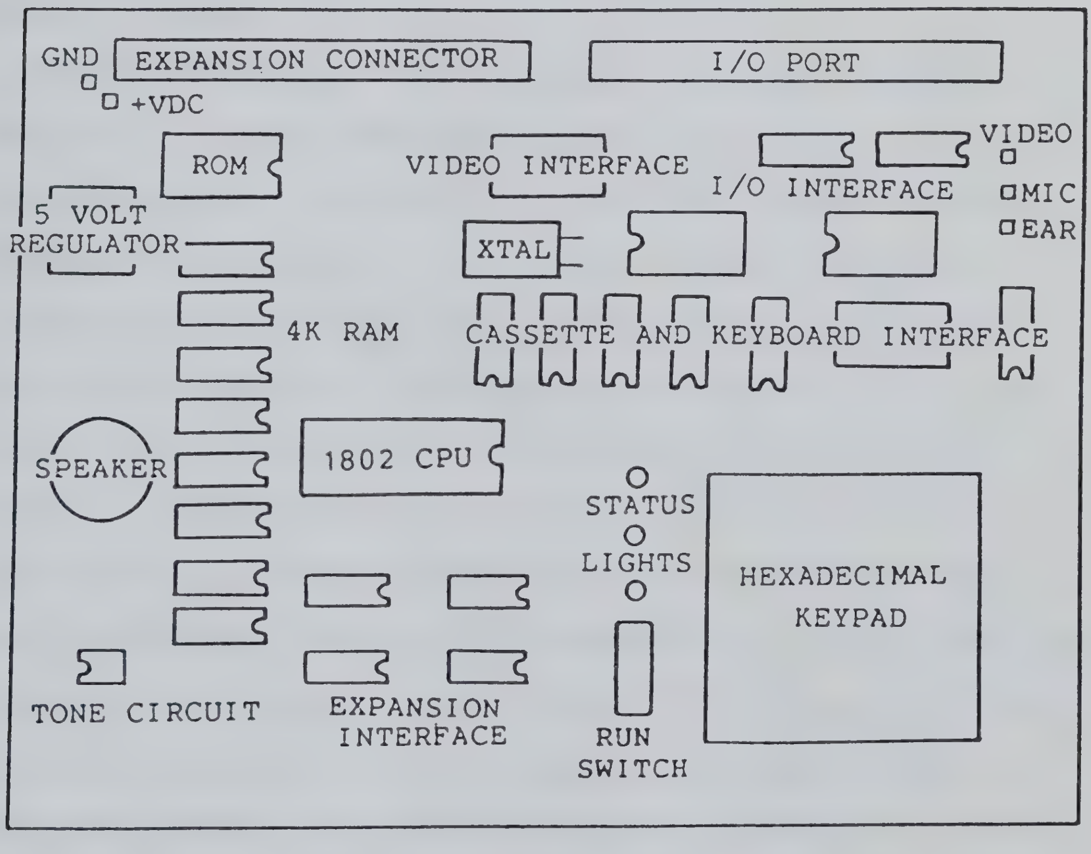


Figure 4.1. The RCA COSMAC VIP-711 microcomputer (a) and layout of its major components (b). The VIP-711 comes with a 5 V power supply; the 5 V regulator and two of the four kilobytes of RAM are optional.

current application. The interface can transfer N pages (256 bytes per page) of data between RAM and tape in 5 + 2.5 N seconds which is almost 100 bytes per second. The first five seconds are used to record a constant tone which signals start of a data block during tape playback. This interface, while just fast enough for the current application, is over three times faster than the Kansas City Standard interface used by many single board and home microcomputers. To obtain a significantly higher transfer rate than that offered by the VIP-711 would require an expensive tape transport or digital recorder.

The VIP-711 must be programmed in COSMAC machine language via the hexadecimal keypad. A detailed discussion of this language is given in the RCA *User Manual for the CDP1802 Microprocessor.*[8] The COSMAC instruction set is efficient and relatively easy to use. There are only 91 instructions compared, for example, to 153 offered by the popular Z-80 microprocessor from Zilog. Although poorly suited to mathematical applications, the COSMAC is well suited to the present control task. The operating system and video interface, provided with the VIP-711, aid in entering, checking and running programs in RAM; unfortunately, there are few provisions to aid in program development and debugging. Such basic editing features as insertion and deletion are absent. Also missing is the option of single step program execution offered by some single board systems.

[8]Available from RCA Solid State Division, Somerville, New Jersey

Consequently, development of software is considerably simplified by access to a COSMAC cross-assembler. This is a program, on a host computer, which will accept COSMAC assembler language mnemonics' and translate them into machine language hexadecimal code. A limited version of the RCA COSMAC cross-assembler is supported by the Computing Services at the University of Alberta and was used in this project.

Programming the VIP-711 requires an understanding of the register structure of the 1802 which is depicted in Figure 4.2. The 8-bit data (D) register, which most manufacturers refer to as the accumulator, is used to hold the input and output of all arithmetic operations. The single-bit data flag (DF) associated with D indicates carries or borrows. The 1802 has sixteen multipurpose, 16-bit registers not found on most microprocessors. Rather than having a special register to act as the program counter (which always contains the address of the next instruction to be executed), one of the general purpose registers is designated program counter by the 4-bit program counter pointer, P. Similarily, the data register pointer, X, is used to point to a register which will contain the address of the next data byte. The 8-bit T register is used to temporarily store the current P and X values when an interrupt occurs. There are four input or external flags

----

'For example, SEQ is used instead of #73 to represent the command to set Q on. The # is used to indicate a hexadecimal number.

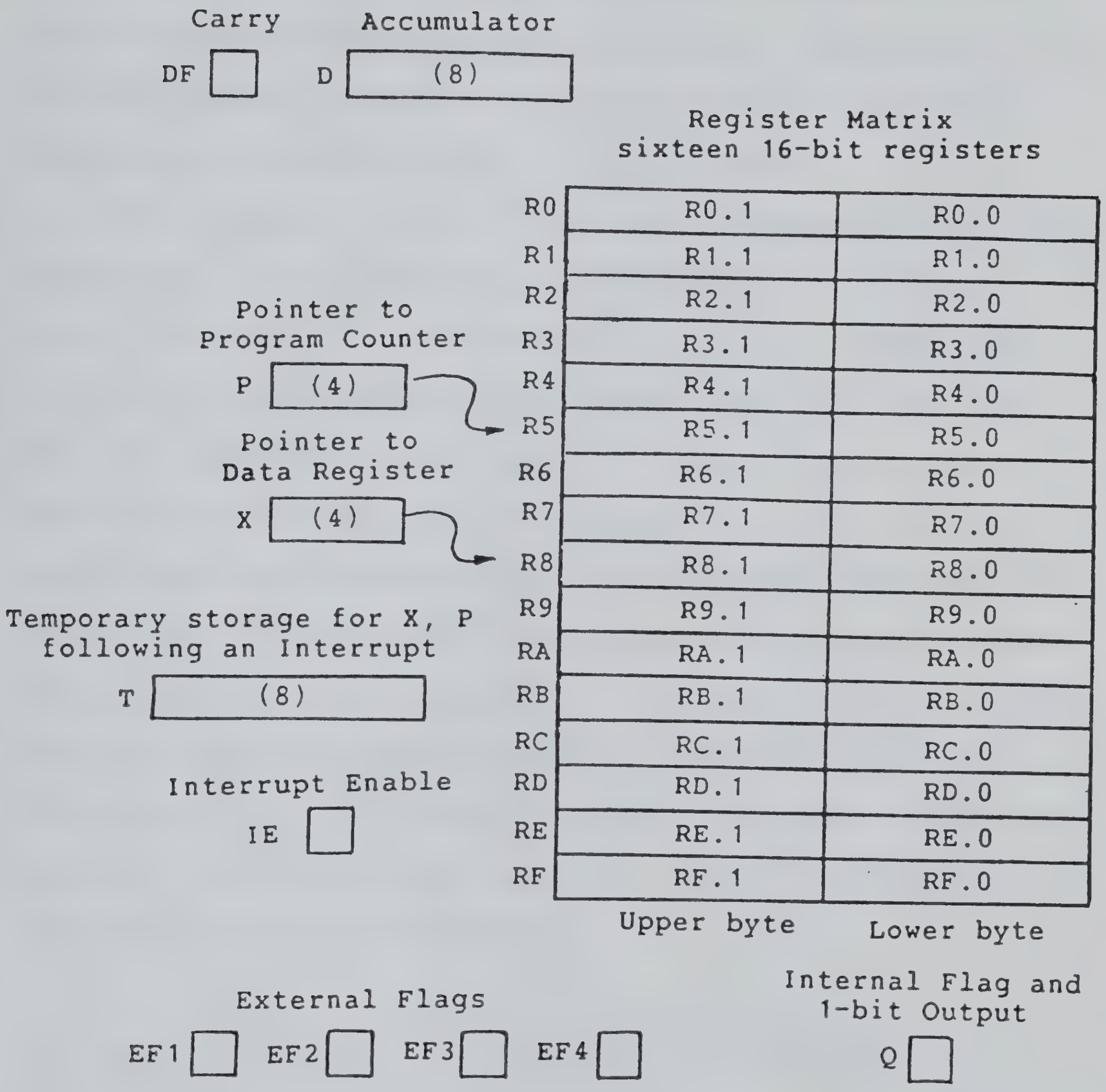


Figure 4.2. The registers in the COSMAC 1802 microprocessor after Peatman (1977).

(EF1 to EF4) to provide information from the outside. Several branching instructions are associated with each of these. There is also a single output bit, designated by Q, which can be turned on or off in software. In the VIP-711, Q is connected to a speaker via a tone circuit. Q is also available, in buffered form, on the I/O port.

The I/O port of the VIP-711 is an important feature not offered on all single-board microcomputers. It includes an 8-bit input port, an 8-bit output port, power connections, an input strobe, plus buffered connections to Q, EF3, and EF4. The input port allows the #6B instruction to read a byte (false= 0 V, true= 5 V) into the data register and the memory location designated by X. An external signal applied to the input strobe (INST) can automatically cause the byte on the input port to be latched. A #68 instruction can then read this byte into memory. The #63 instruction places the contents of D on the parallel output port, OUT0 to OUT7, in order to control external equipment. The pin designations of the port are given in Table 4.1.

TABLE 4.1 VIP-711 I/O PORT CONNECTIONS

PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL
A	IN0(LSB)	K	INST	U	OUT6
B	IN1	L	EF4	V	OUT7
C	IN2	M	OUT0(LSB)	W	Q
D	IN3	N	OUT1	X	EF3
E	IN4	P	OUT2	Y	+5V
F	IN5	R	OUT3	Z	GND
H	IN6	S	OUT4		
J	IN7	T	OUT5		

4.2 Analog-to-Digital Conversion

An Intel 1H4108, eight-channel multiplexer is used to pass the different signals to a Teledyne Semiconductor 8700CJ ADC. The 8700 is fairly typical of numerous ADC devices currently available. It is an 8-bit CMOS chip housed in a 24 pin package, capable of up to 800 conversions per second. Cost was about $15. Only a negative reference voltage, dual power supply, and a few passive support elements are required for operation. Any desired positive input range can be selected (nominally 0 to 5 V here). The circuit, which closely follows the recommendations of the manufacturer, is shown in Figure 4.3. The positive power supply is provided by a standard 5 volt regulator (7805). Negative 5 volts is obtained from this with a unit-gain inverting operational amplifier (741) circuit. An LM336 voltage reference chip (+2.499 V for 0°C to 70°C) and an inverting operational amplifier provide an adjustable negative voltage reference.

There are several advantages to chosing an eight bit device. They are much cheaper than 10- or 12-bit units of the same speed and they interface easily with an 8-bit microcomputer. Byte-size words can be manipulated much more efficiently and stored in less space than longer ones. The drawback is resolution. The full-scale resolution of an 8-bit ADC is 0.4%. While this sounds respectable, if the signal fluctuations only cover one tenth of the available range then the resolution is at best 4%. For the measurement

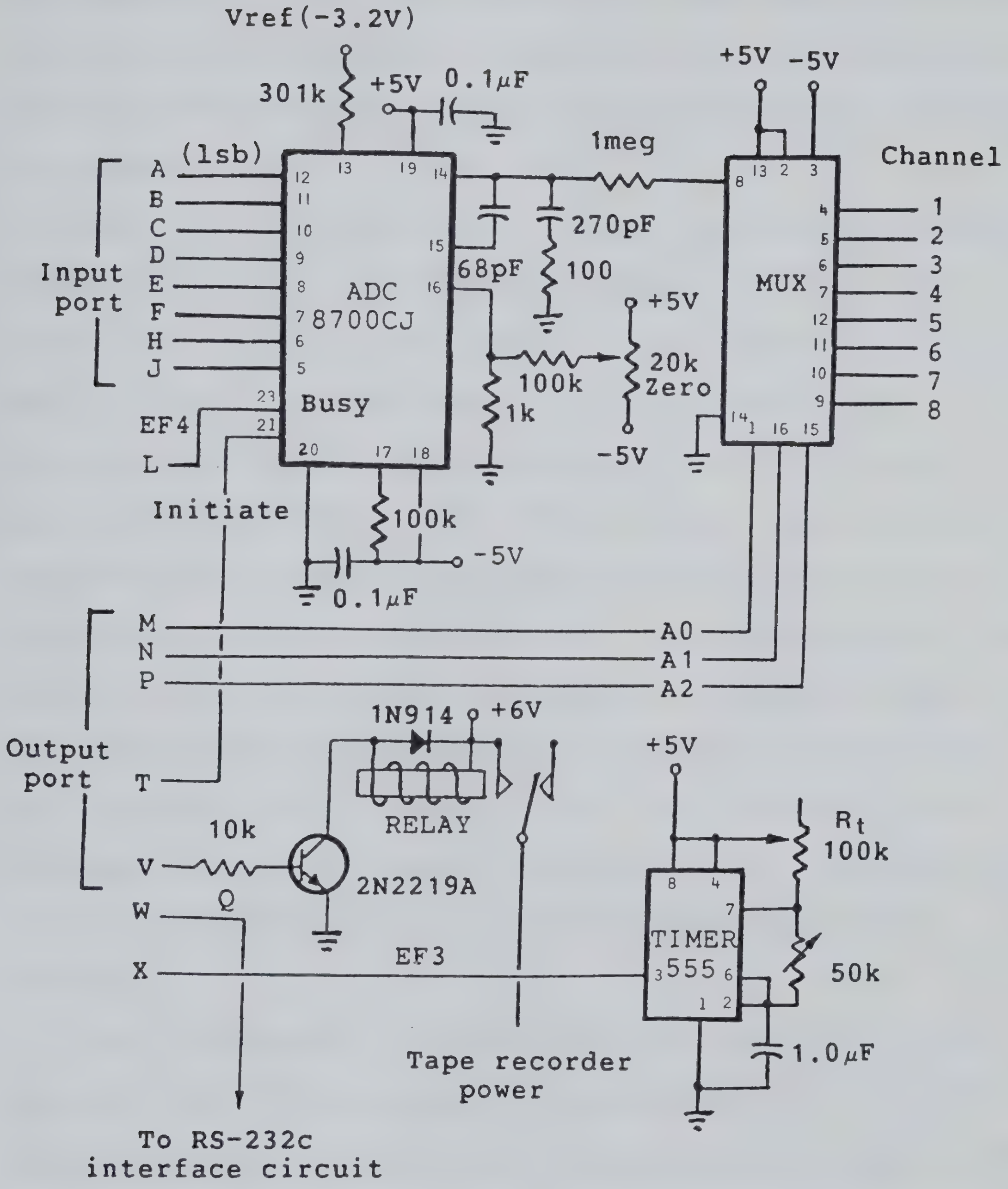


Figure 4.3. Connections to the VIP-711 I/O port. In the actual circuit, various values of R_t are selected by a switch (labeled CLOCK) to provide pulse rates of 5, 10, 15 or 20 Hz.

of atmospheric turbulence this points out the importance of offsetting the large mean values and amplifying the fluctuations so that they ultilize the available digitizing range as fully as possible. If this is done properly there would be little benefit, given sensor limitations, in using more bits.

The I/O port of the VIP-711 makes interfacing the ADC and MUX very simple. The complete circuit is shown in Figure 4.3. The output of the ADC connects directly into the parallel input port (IN7-IN0). The 8-bit output port provides the control signals required. Its three least significant bits (OUT2, OUT1, OUT0) select which multiplexer channel is to be sampled. Binary settings of 000 through 101 are used to designate channels one to six respectively. The OUT5 bit instructs the ADC to commence a conversion when set high. The MSB of the output port byte, OUT7, controls the tape recorder via a low-power relay. When this bit is low (the usual state) the cassette recorder is on. When the OUT7 is high, the cassette is off and the relay draws about 100 mW.

The two external flags available on the I/O port are also used. EF3 is connected to the external clock circuit in Figure 4.3 which is used to set the sampling frequency. EF4 is connected to the BUSY pin of the ADC thus enabling the computer to keep track of the status of the conversion process.

A flowchart of the digitizing process is given in Figure 4.4. The collection process begins when EF3 detects the rising edge of a clock pulse. This initiates a conversion scan intended to simulate simultaneous sampling. First, an #80 (binary 1000 0000) is placed on the output port which selects the first channel but does not initiate a conversion. After a short delay (18 μs) to allow the multiplexer time to settle down, an #A0 (binary 1010 0000) starts the conversion. Both the #80 and #A0 keep the cassette recorder power turned off. Once the conversion is started, the VIP-711 continuously polls EF4 which is connected to the ADC Busy pin. When the conversion is complete EF4 is set high by the ADC and the computer immediately latches up the byte on the input port and stores it in the next available memory location. The next multiplexer code, normally #81, is then sent out followed, as before, by #A1. This cycle continues until all active channels have been sampled at which point the computer resumes polling EF3. The time per conversion is approximately 1.25 ms, hence a six channel scan takes about 10 ms. This is an order of magnitude faster than the typical sampling rate of 10 Hz thus providing an adequate approximation to simultaneous sampling.

The data collection proceeds as above until the RAM buffer is filled. The first two pages of RAM (#0000 to #01FF) are used to hold the data collection software. The 80 bytes at the top of RAM are used by the operating system,

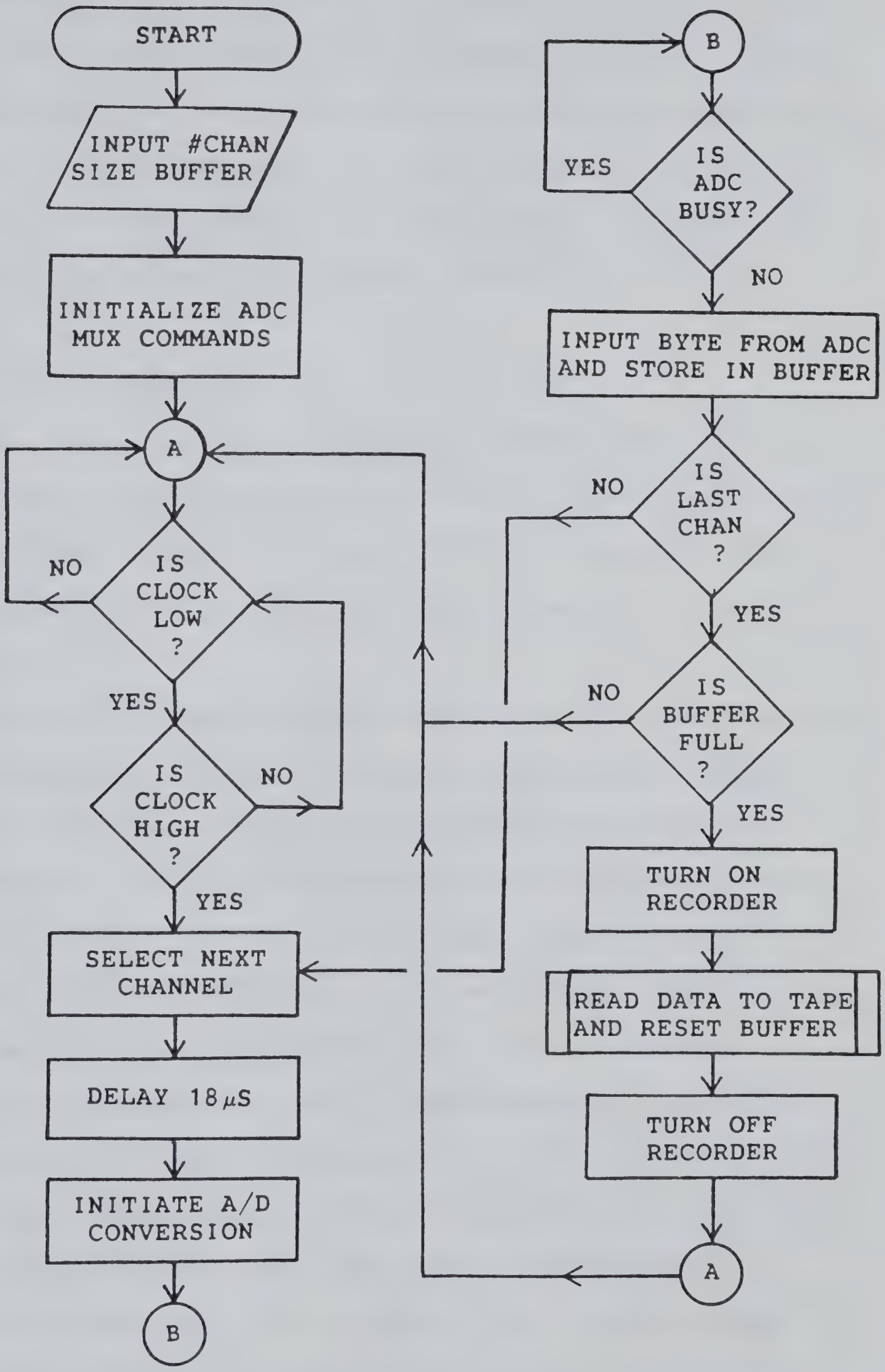


Figure 4.4. Flowchart of analog to digital conversion.

hence the last page of RAM is not included as part of the buffer. This leaves a total of 13 pages (3328 data values) for the buffer which, of course, must be divided among the number of active channels. This can be specified in the program, as in the software in the Appendix, entered as input via the keypad for greater flexibility.

4.3 Writing to Tape

When the available semiconducter RAM is filled, the collection process is halted and data is transfered to tape. Sampling four sensors at 10 Hz will fill up about eight pages of RAM per minute so this dumping must occur quite frequently.

By using the subroutines contained in the RCA ROM based Operating System the data collection process was implemented with only moderate software development. In order to do this, however, it was first necessary to disassemble and decipher the VIP-711 Operating System. A memory map of this software is given in Table 4.2. A documented listing of the VIP cassette write software is given in the Appendix. Basically, the software, via Q, generates one cycle of a square wave at 2 kHz to represent a low-bit and one cycle at 0.8 kHz for a high-bit. The interface hardware rounds these into crude sines waves for the audio tape recorder.

In the data collection software, R3 is initialized to the address (#8091) thus allowing the RCA write-to-tape routine to be called by making R3 the program counter. In

TABLE 4.2. RCA OPERATING SYSTEM MEMORY MAP

Address	Function Starting at given address
8000	Locate top of RAM (from #01FF)
8022	Run program or enter operating system monitor
8027	Operating system intializations
806E	Keypad select operating system function
8091	Tape write routine
80C2	Tape read routine
80EF	Memory read routine
80F3	Memory write routine
8110	Digit display for video output
8143	Video interrupt routine
816F	Subroutine to write a bit to tape
8182	Subroutine to read a bit from tape
8194	Hex keypad input subroutine

addition, RC is initialized to #816F, the starting address of the the RCA subroutine to read single bits to tape. This subroutine terminates with an unconditional branch to location #816E where P is reset to R3 thus returning control to the exit point in the tape-write rountine. Doing this from #816F leaves RC with the starting address of the subroutine (816E+1) for future calls. This is the most common method of calling COSMAC subroutines. The tape-write routine is not, however, such a subroutine as it terminates in a closed loop; thus, an escape route is needed in order to use this software. In addition, values normally supplied manually via the hexpad must be generated in software. Fortunately, just prior to falling into this loop, a video display subroutine is called by making R4 the program counter. Since a video display is not needed during the data

collection process, this subroutine call can be used as way out of the operating system. By making R4 the main program counter of the data collection program the write routine will return control almost like an ordinary subroutine. The only difference is that RC is left pointing to the last line of the routine, rather than the first, after each call. Thus R3 must be reinitialized before each call, a small price to pay in return for avoiding the need to reproduce the tape interface software in RAM.

4.4 Field Operation

The VIP-711 comes with a light, plastic case which is not suitable for outdoor operation. Figure 4.5 shows the plexiglass enclosure constructed to house the VIP-711, tape recorder, voltage regulators, RF modulator and relay circuit. In the field, this unit sits on top of the APU; the two units are connected by a single ribbon cable. An external 12 V lantern battery must also be connected to the microcomputer unit (MCU). During initial experiments, the MCU was allowed to draw power from the APU. This was found to be unadvisable, however, since the power consumption of the computer (about 3 W) is much greater than that of the APU. In fact, to extend battery life, the power to the MCU should not be connected until the analog gain and offset adjustments have been made.

When ready to begin data collection, the data collection software must be loaded from cassette tape into

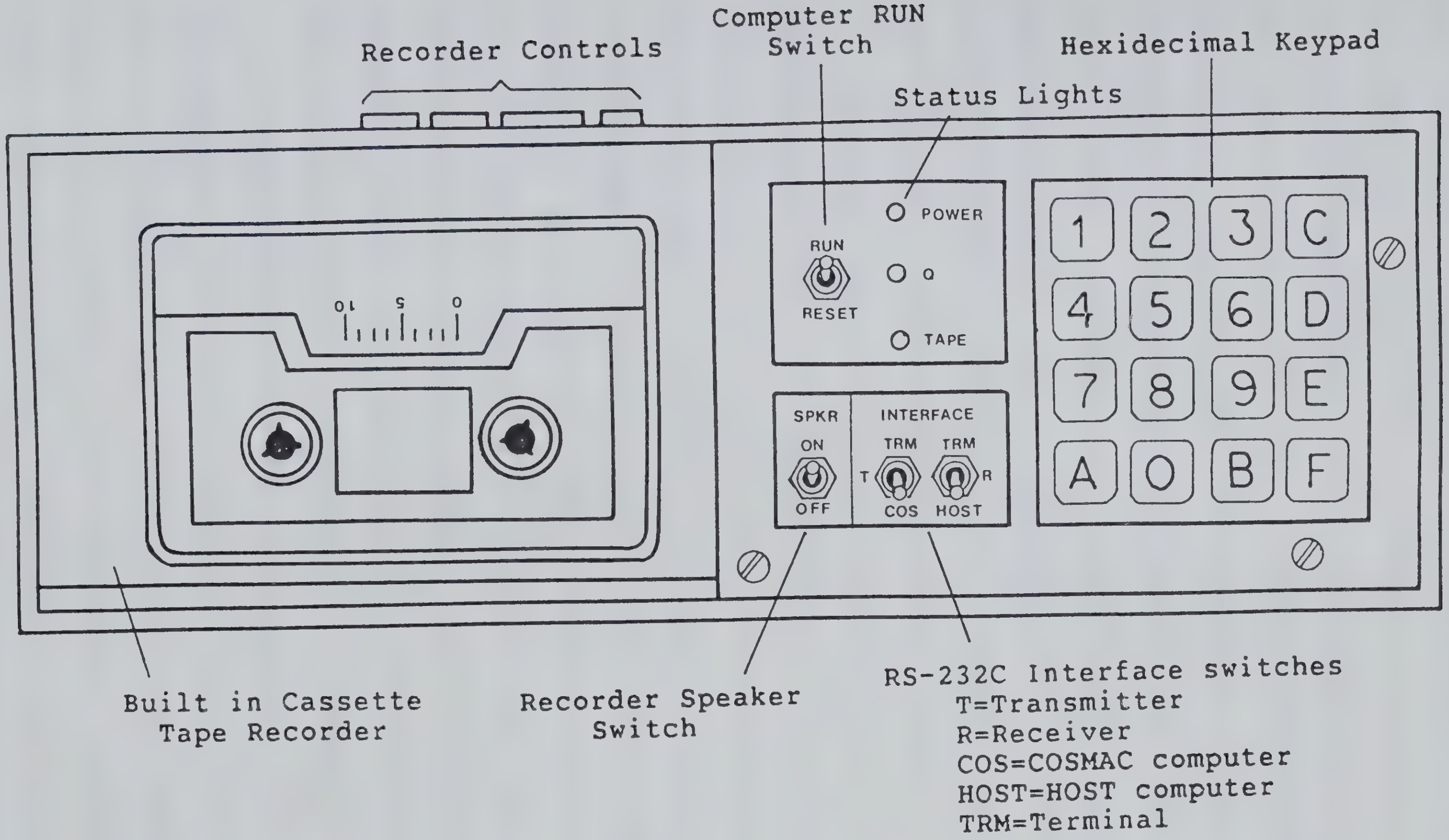


Figure 4.5. Front view of the microcomputer unit (MCU). The VIP-711, cassette tape recorder, RF modulator, relay circuit and RS-232C interface circuitry are housed in a single plexiglass unit. The RS-232C interface and control switches are discussed in Chapter 5.

[illegible]

the first two pages of RAM. This is done by placing the cassette into the tape recorder and moving to the start of the audio code. The VIP-711 operating system is then selected by setting the microcomputer to RUN while holding the C key depressed. To load a two page program into memory starting at location #0000, the sequence of keys 0000B2 are pressed and the recorder is turned to PLAY. The tape light goes out when the program is loaded and the software cassette should be removed and replaced with a blank cassette for data collection. To run the program the microcomputer is RESET then switched to RUN.

Numerous versions of data collection software were experimented with; the program given in the Appendix was found to be the most useful. Upon running the program, execution pauses to wait for two input parameters. The first is the number of channels to be used, taken sequentially from T, W, U, V, X, and Y. While this may seem somewhat restrictive, it ensures no confusion about the contents of a tape during playback. The second input parameter is the number of pages per channel to be allotted for buffering. The software checks both inputs for illegal values.

Once the two parameters are supplied from the hexpad, the program waits for the APU clock to be switched to the desired sampling frequency. At this point sampling commences and the tape recorder should be turned to RECORD (power is off). The microcomputer fills the buffer, turns on the recorder, writes to tape, turns the recorder off and repeats

this cycle until the computer is RESET. Thirty minutes of tape will hold about 200,000 bytes, which typically represents 2 or 3 hours of data.

The only difficulties encountered during the testing of the prototype system were caused by low batteries. Symptoms of this occurence included software crashes and failure of the relay to operate correctly. New 12 V lantern batteries lasted about 15 hours. When using old batteries, the system must be monitored closely. An audio signal which is generated by the VIP-711 during tape transfers proved to be a useful indicator of proper operation.

4.5 Continuous Data Collection

The major shortcoming of the prototype data collection scheme, gaps in the data which occur during transfers to tape, could be eliminated with improved software. The only difficulty involves the timing done by the microprocessor as it writes individual bits to tape; the microprocessor can not be disturbed during this process which makes it difficult to use the COSMAC interrupt capabilities. However, since it takes about 1 ms to write a single bit to tape, it would be possible to poll the flag connected to the clock a thousand times a second to determine if an A-to-D conversion must be started. Since the clock pulse stays high for more than 1 ms, is should be possible to implement continuous data collection with the prototype system by simply modifing the collection program. It would become necessary, however,

to include the tape write routine in RAM although the RCA subroutine which writes individual bits to tape could probably be used directly from ROM.

The envisioned software would use two buffers, say, one page per channel. Data would be stored in buffer A until it is filled then into buffer B. While B fills, buffer A would be written to tape. Clearly, if no data are to be lost, the sampling rate is constrained by the requirement that one buffer can be written to tape more quickly than the other can be filled. Using one page per channel and sampling at 10 Hz gives 25.6 seconds. For a dedicated tape-write this is enough time to save eight pages (2 kilobytes) of data. Assuming the modified write-to-tape routine is not significantly slower than the present implementation, it should be possible to collect six or seven channels of continuous data at 10 Hz.

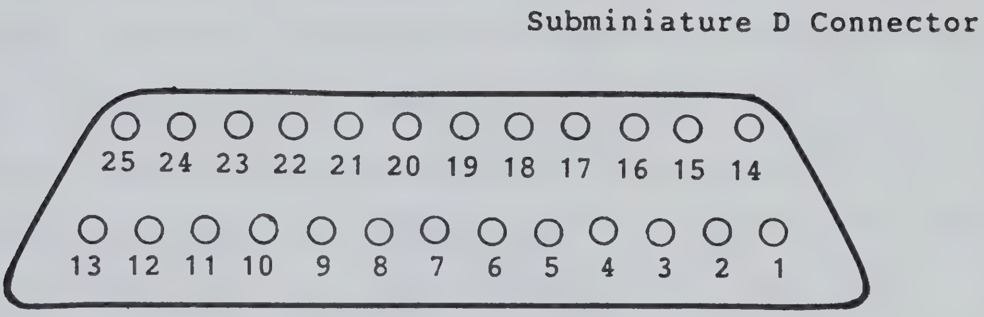
Chapter 5

TRANSFERRING DATA TO A HOST COMPUTER

5.1 A Serial interface

The data stored on cassette tape must be transferred to a host computer before any serious processing can begin. The host computer available at the University of Alberta is an AMDAHL 580/5860 operating under the Michigan Terminal System (MTS). The University's Computing Services supports a large number of remote terminals connected to the computer by modems (Gandalf Data Sets) capable of serial, full duplex data transmission at speeds ranging from 110 to 9600 baud. A Decwriter terminal, normally operating at 600 baud, was available for use in this project. The connection between the terminal and its modem is via the widely used RS-232C standard (see Figure 5.1) which defines interconnect signals, voltages, connector type and pin assignments. Information sent to the AMDAHL must be ASCII (American Standard Code for Information Interchange), with odd parity. Thus, in order for the VIP-711 to use the serial communication link, data must be retrieved from cassette tape, translated from binary to ASCII code and transmitted, one bit at a time, in RS-232C compatible signals. Almost all of this could be done in software. The only hardware additions concerned the RS-232C hook up.

The RS-232C interface requires that a false bit be in the range +5 V to +15 V, while a true bit is anywhere from



1.Protective Ground
2.Transmitted Data
3.Received Data
4.Request to Send
5.Clear to Send
6.Data Set Ready
7.Signal Ground
8.Data Carrier Detect
9...14.Not Used

15.Transmitted Bit Clock Internal
16.Not Used
17.Received Bit Clock
18,19.Not Used
20.Data Terminal Ready
21.Not Used
22.Ring Indicator
23.Data Signal Rate Selector
24.Transmitted Bit Clock
25.Not Used.

Figure 5.1. The RS-232C standard. The pin definitions above refer to the terminal side of the connector. In the present application only pins 2, 3, and 7 are of interest. A high bit must be in the range -5 to -15 V, while a low bit is anywhere from 5 to 15 V.

-5 V to -15 V. The buffered Q line on the VIP-711 I/O port, used to produce the serial output, generates 0 or 5 V signals. While special integrated circuits are available to transform such signals to RS-232C levels, the conversion can be easily done with more common components. The noninverting operational amplifier configuration shown in Figure 5.2 was used here. The circuit transforms 0 V to -10 V and 5 V to 10 V which seems to be the opposite of what is required. This is done because the AMDAHL requires a high signal when no data is being sent but Q is normally low. Using the circuit shown, Q must be turned on to represent a false bit and off for a true bit. This is easily done in software.

The transformed serial output must be connected to the receive line on the modem. However, rather than simply replacing the normal terminal with the VIP-711, the connections shown in Figure 5.2 leave the usual terminal host connections intact while allowing the VIP-711 to tap into or drop out of the communication link. This allows, for example, the standard terminal to be used to log onto the system before the VIP-711 is engaged to send data. At the end of a data dump, control can be easily passed back to the terminal to allow manipulation of the data file without disconnecting the VIP-711. The interface switches on the front panel of the microcomputer enclosure (see Figure 4.9) allow transmission of information from the microcomputer to the terminal and the host as well as the normal terminal to host link.

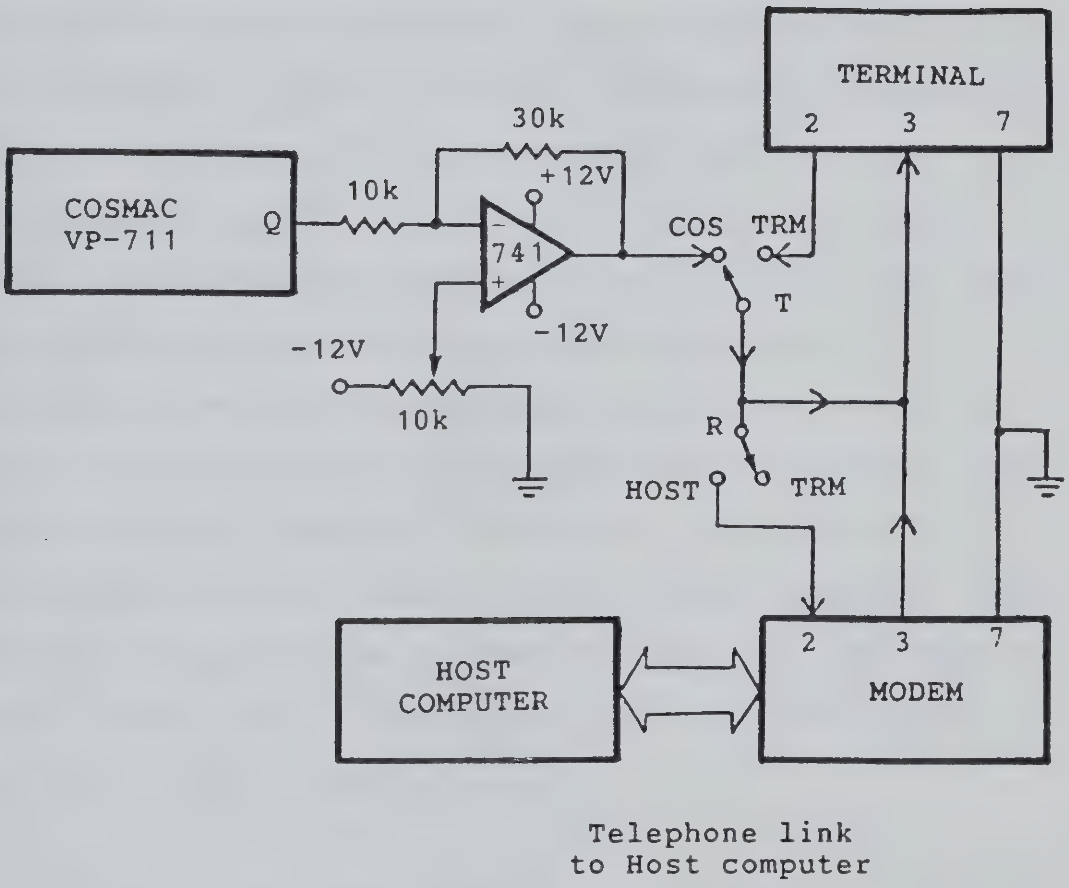


Figure 5.2. Hardware interface between the VIP-711 and a host computer and terminal. The serial interface was operated at 600 baud to allow simultaneous transmission to both the terminal and the host computer.

5.2 Reading from Tape

Reading data from the cassette under program control is similar to the writing of data previously discussed. The VIP hardware roughly converts the audio signal back into a square wave which is monitored by EF2. A bit begins when EF2 goes high and the VIP software determines whether the bit is true or false by timing from the rising edge to the falling edge. R3 serves as the program counter for the read routine which begins at #80C2. This time RC must be intialized to #8182, the subroutine which reads in individual bits. Again R4 is used as the main program counter so that it is possible to escape from the operating system. The total number of pages to be read from RAM must be placed in RE.0. In the software given in the Appendix, this value is calculated from the number of channels and pages per channel which are supplied from the hexpad along with the number of channel scans to be printed per line. The output buffer begins at location #0200.

5.3 Binary to ASCII Conversion

After filling the RAM buffer the VIP-711 must begin sequentially sending ASCII data to the AMDAHL. ASCII is designed to handle alphanumeric information and, as a result, is rather inefficient for transmitting pure numeric data as must be done here. For example, the decimal number 234 is represented by the single binary byte 1110 1010 (#EA). To transmit this value in ASCII requires a byte for

each decimal character plus a byte for the ASCII space character.

The conversion from binary to ASCII is aided by the way the ASCII code is set up. Adding 30 to a number between 0 and 9 gives the resulting ASCII hex code. Therefore, the problem is to break the binary representation down into its three decimal digits. The procedure, depicted in the flow diagram in Figure 5.3, begins by subtracting decimal 100 (#64) from the data byte. If the remainder is less than zero, the first digit is zero. If not, an additional subtraction is performed to find out if this digit is a one or a two. The remainder of the last successful subtraction is then subjected to a generalized version of this procedure where 10 (#0A) is successively subtracted until the tens digit is found. The ones digit is automatically produced as the remainder of the the tens loop.

5.4 Parallel to Serial Conversion

As each ASCII character is generated it must be serially transmitted to the host. In addition to the seven ASCII bits associated with each digit, a start bit, odd parity bit, and stop bit must be transmitted. Inexpensive monolithic integrated circuits called UART's (Universal Asynchronous Receiver Transmitters) are available to provide this formatting. Here, however, it was implemented in software at no extra cost.

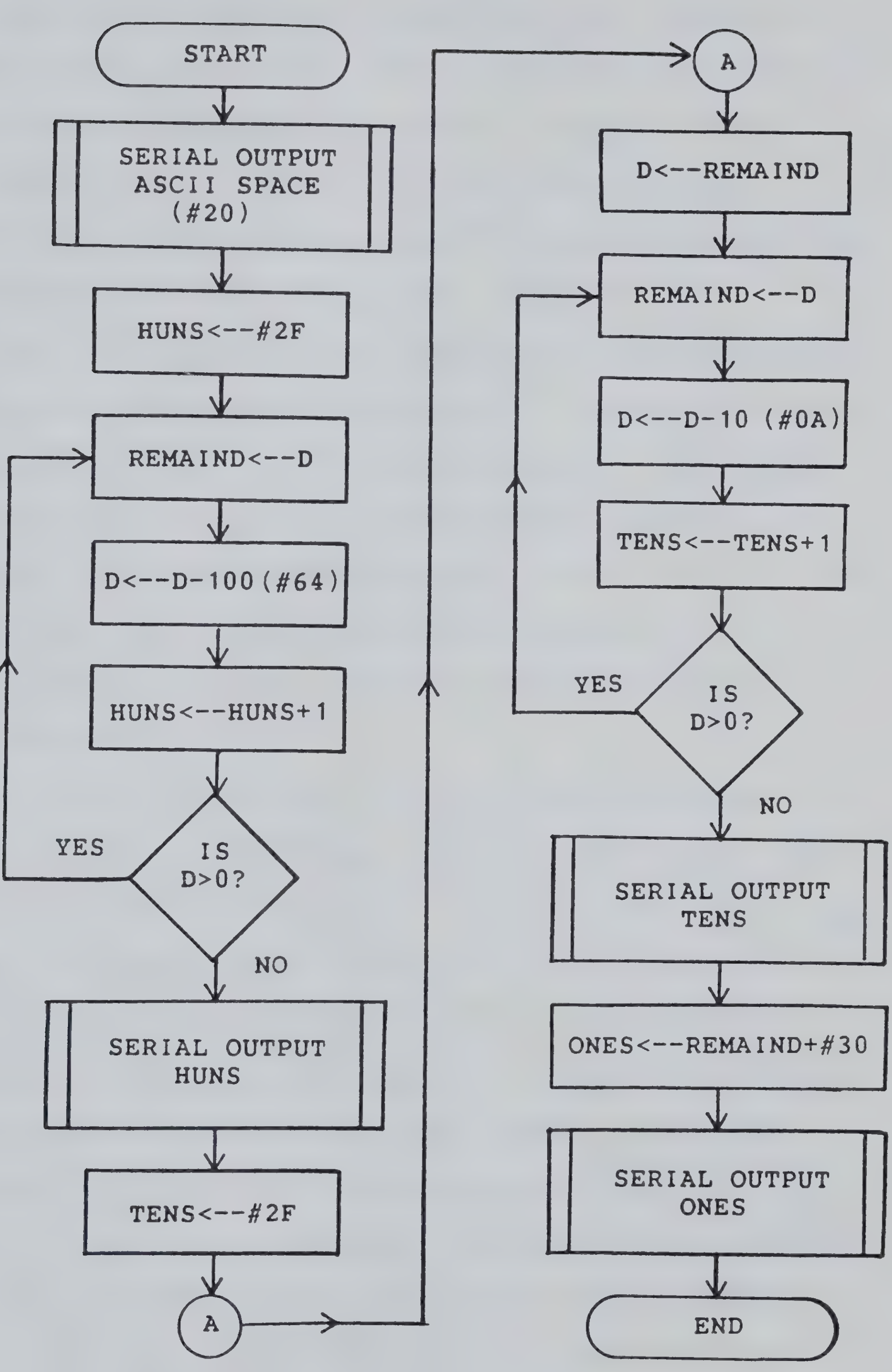


Figure 5.3. Flowchart for Binary to ASCII conversion. The Flowchart starts with the binary data byte in the data (D) register. The hundreds (HUNS), tens, and ones digits are sequentially determined and transmitted.

A true bit is generated by setting Q low (false bit requires Q high) and falling into a software delay loop. For 600 baud the necessary delay is about 1.6 ms.[10] In the actual software, the time of the delay involves some overhead due to additional instructions executed before and after entering the loop. Since the number of these additional instructions varies depending on where the delay subroutine is called from, the number of iterations is sent as an input parameter to the delay subroutine. Values were determined with the aid of an oscilloscope, however, the setting was not particularly critical. Fine tuning would be more important for higher baud rates. It seems likely that 1200 or 2400 baud transfers could be implemented using the delay loop approach.

The serial transmission of a digit, flowcharted in figure 5.4, begins by sending a low start bit to the AMDAHL. Then, with the data byte in the D register, each bit of the data byte is sequentially shifted into DF. By testing to see if DF is a one or zero, Q can be set to the corresponding value. One of the scratchpad registers is used to generate the parity bit. Intially zero, each time a high bit is encountered this register is incremented by one. When all seven bits have been transmitted, the LSB of this register

----

[10]The simplest COSMAC delay loop has the following form;

```
 loop    DEC Rn;    Decrement Rn
         GLO Rn;    Place Rn.0 in the D register
         BNZ loop;  Goto loop if Rn.0= 0
```

where the intial value of Rn determines the delay time. Each of the instructions requires about 9 μs to execute giving a 27 μs delay per loop. This requires 60 (#3C) repetitions for 1.6 ms.

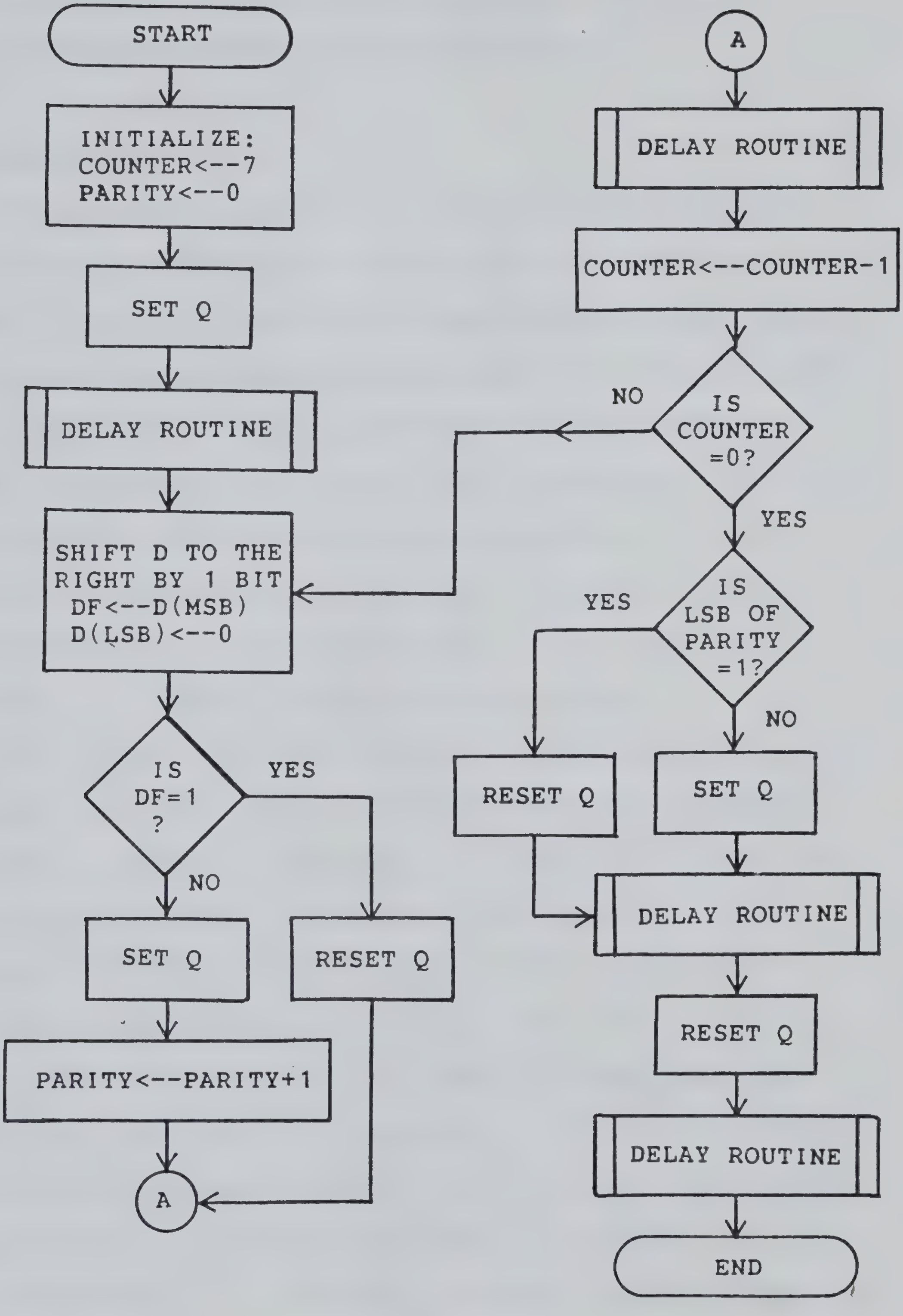


Figure 5.4. Flowchart for parallel to serial conversion. For a high bit (DF=1) it is necessary to reset Q. For a low bit (DF=0) Q is set on.

will contain the required odd parity bit. This is transmitted and followed by a high stop bit.

5.5 Operation

With the microcomputer connected into the terminal modem link, and interface switches set at T= TRM and R= HOST, the user logs onto the host system in the usual manner. For the AMDAHL/MTS system used with the prototype, it is important to issue a %PRENT=ON command before attempting any data transfers. This command activates input line buffering by the front end processor of the host system. If this feature is not used, occasional characters will be lost when the computer, due to its time sharing structure, is unable to respond immediately.

The serial interface program is loaded from tape (same procedure as used to load the data collection software in Section 4.5) and the data tape is placed in the recorder. The microcomputer is switched to RUN. In the prototype software, the number of channels, number of pages per channel, and the number of channel scans to be stored per line of the host computer file must be entered via the hexpad. At this point the T-switch is set to COS (COSMAC), the cassette is turned to PLAY, and the last key pressed is pressed again to initiate the transfer process.

At 600 baud, the transfer of data is rather slow. The conversion from binary to ASCII results in four times more characters to be transmitted than were originally saved.

However, it is convenient to use 600 baud, since it allows easy switching between the VIP-711 and the terminal, as well as a simultaneous hard copy output. Parity errors are occasionally detected when reading data from tape to RAM. When this occurs, the VIP-711 link with the computer is automatically broken. It is then necessary to rewind the tape to the start of the bad block and reread it. In many cases the parity error does not reoccur. If the error does persist, it can be ignored (likely just a single bit) or the entire block can be skipped.

Chapter 6

DATA ANALYSIS

6.1 Experiments

Considerable turbulence data were collected during a series of initial experiments intended primarily to test the performance of the various components of the prototype data acquisition system. High-level processing was performed on some of these data in order to compare computed values with those found by others over flat, uniform terrain thus providing an additional, although limited, check on the operation of the system.

Most of the experiments were conducted at the University of Alberta Research Farm located about 10 km south of Edmonton. A permanent, open constuction, 10 m tower is located at this site which is a modest approximation to horizontal homogeneity. Experiments considered here were restricted to days when the wind was predominantely from the the west, as this provided the most uniform, unobstructed approach to the tower. Even in this direction, however, the land sloped downwards at a rate of about one in thirty, and had a fetch of only about 300 m. Ground cover around the tower was short grass. About 50 m to the west this changed to a barley crop which was 30 cm high in early experiments, but cut to short stubble in later ones (those in October).

Typically, turbulence measurements were made at about 5 m. Attempts were also made to obtain supporting wind and

temperature measurements; however, the difficulty of obtaining reliable profile measurements should not be underestimated. In most cases, the available equipment for these measurements, Rimco impulse cup anemometers and fine-wire chromel-constantan thermocouples, were inadequate for resolving the small gradients encountered. Consequently, profile estimates of stability and surface layer parameters are quite crude. The poor profile data, site imperfections, and the low frequency cut-off imposed by the prototype data acquisition software make it unproductive to compare profile and covariance flux estimates. Here, therefore, the focus is on the spectral analysis of turbulence which the collection of raw data makes possible.

6.2 Sonic and Platinum wire Power Spectra

The collection of data in segments containing multiples of 256 points is well suited to spectral analysis via the Fast Fourier Transform (FFT) method which requires time series of length 2^m (m an integer). Typically, transforms were performed on segments containing L= 256, 512, or 1024 raw data points. This length determines the lowest frequency which can be reliably resolved ($1/L\Delta t$ where Δt is the sampling period) by the subsequent analysis. Each segment was detrended with a least-square-fit and its ends (10%) were tapered with cosine bells prior to transforming as described by Bingham et al. (1967). The complex Fourier coefficients were calculated with a standard FFT program

then squared, summed, and scaled to provide modified peridograms.[11] Several consecutive peridograms were averaged to provide a relatively stable estimate of the true spectrum.

Following the usual convention in micrometeorology, log{nS(n)} is plotted against log f. Such a graph, unlike one of log{S(n)} vs log f, maintains an area-variance equivalence. Figure 6.1 shows a typical vertical velocity spectrum plotted in both ways. The spectrum is the result of averaging 20 consecutive peridograms covering a period of about 15 minutes. While the large scatter implies the need for longer averaging periods, the strong influence of diurnal variations restricts such an approach; rarely can stationarity be assumed for more than 30 minutes in the atmosphere. Following current practise, a smoother picture is obtained by block averaging the spectral estimates over logarithmic intervals to give about 20 equally spaced values on a logarithmic frequency scale. On such a graph, the high frequency values involve averaging over many more estimates than do lower values and, therefore, are more stable. In particular, not too much confidence can be placed in the first few low-frequency estimates.

Figure 6.2 shows the block-averaged logarithmic spectra of sonic vertical wind, nSw(n), and platinum wire

---

[11]The standard FFT routine distributes the variance over $L=2^m$ estimates, of which only the first L/2 are unique. Therefore, the spectra are scaled by multipling by 2 and dividing by Δn such that $\sigma^2 = \Sigma_{i=0}^{L/2} S_i \Delta n$ in keeping with the definition for power spectra used in micrometeorology, $\sigma^2 = \int_0^\infty S(n)dn$.

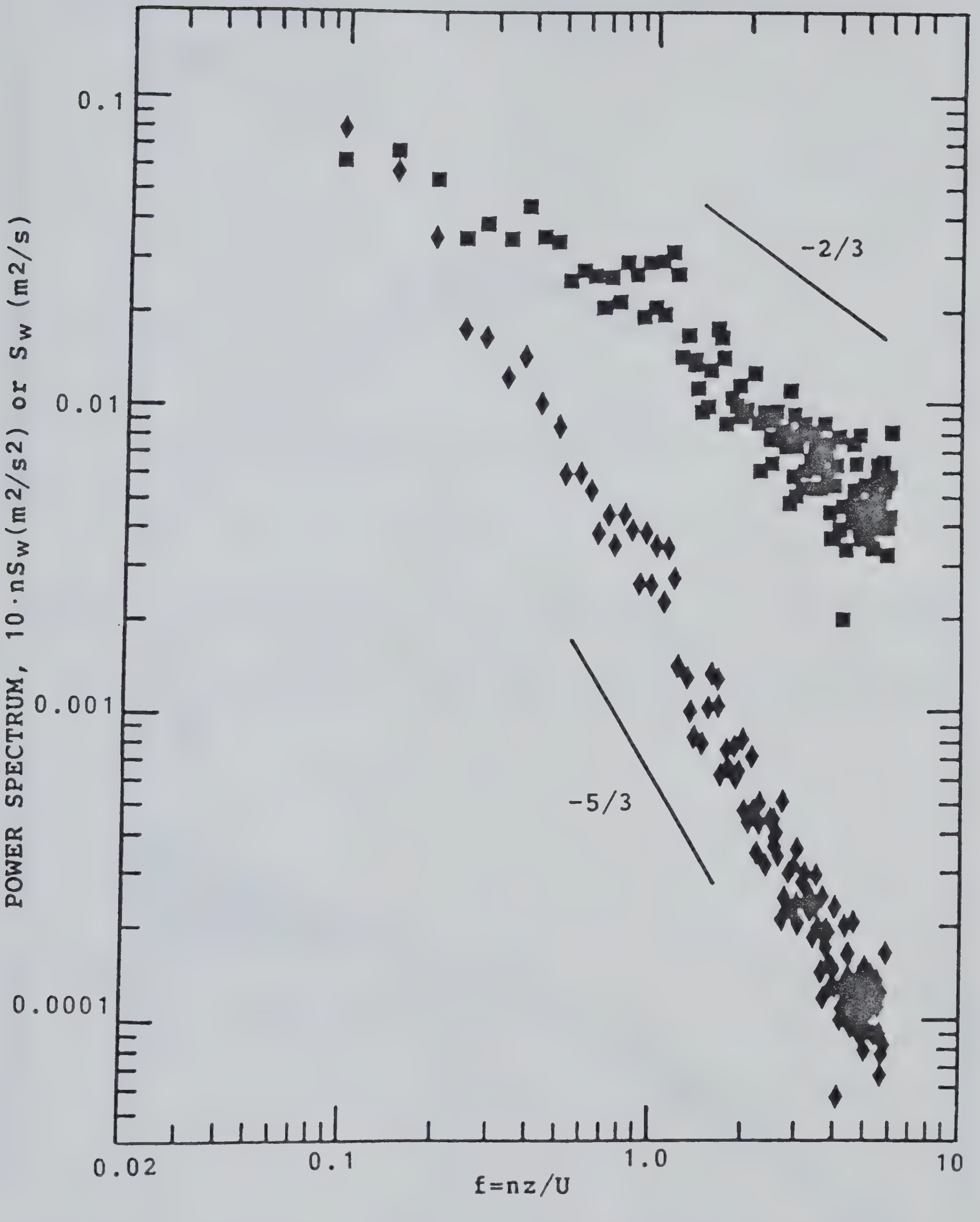
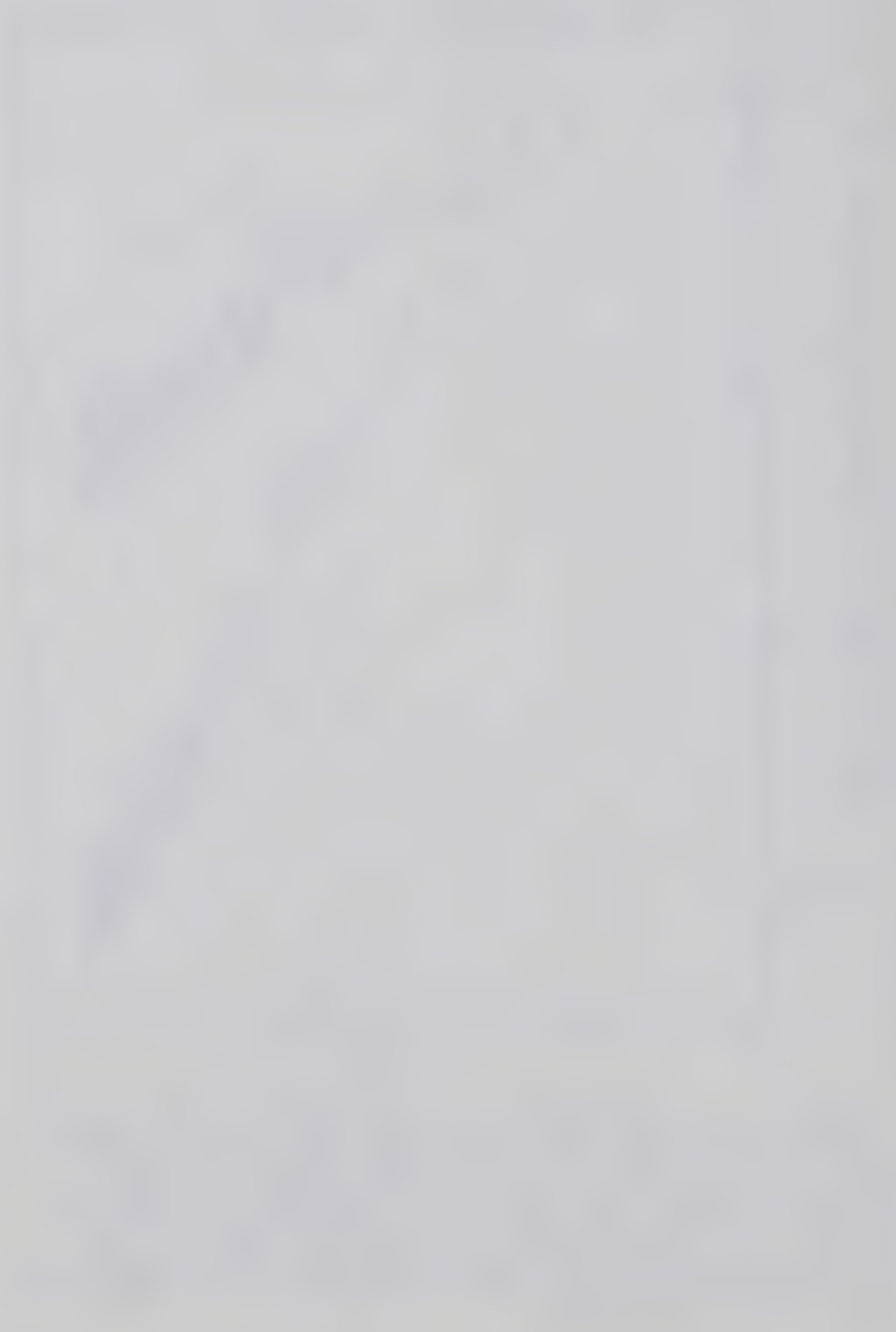


Figure 6.1. A typical vertical wind spectrum obtained from 15 minutes of 10 Hz sonic anemometer data. Twenty segments, each containing 256 values were Fast Fourier transformed to give periodograms with 128 estimates, each which were averaged together. The lower plot (diamonds) is log{$S_w(n)$} versus log f and therefore is expected to follow a -5/3 slope in the inertial subrange. The upper plot (squares) is the logarithmic spectra, log{$nS_w(n)$} versus log f and should thus follow a -2/3 slope.



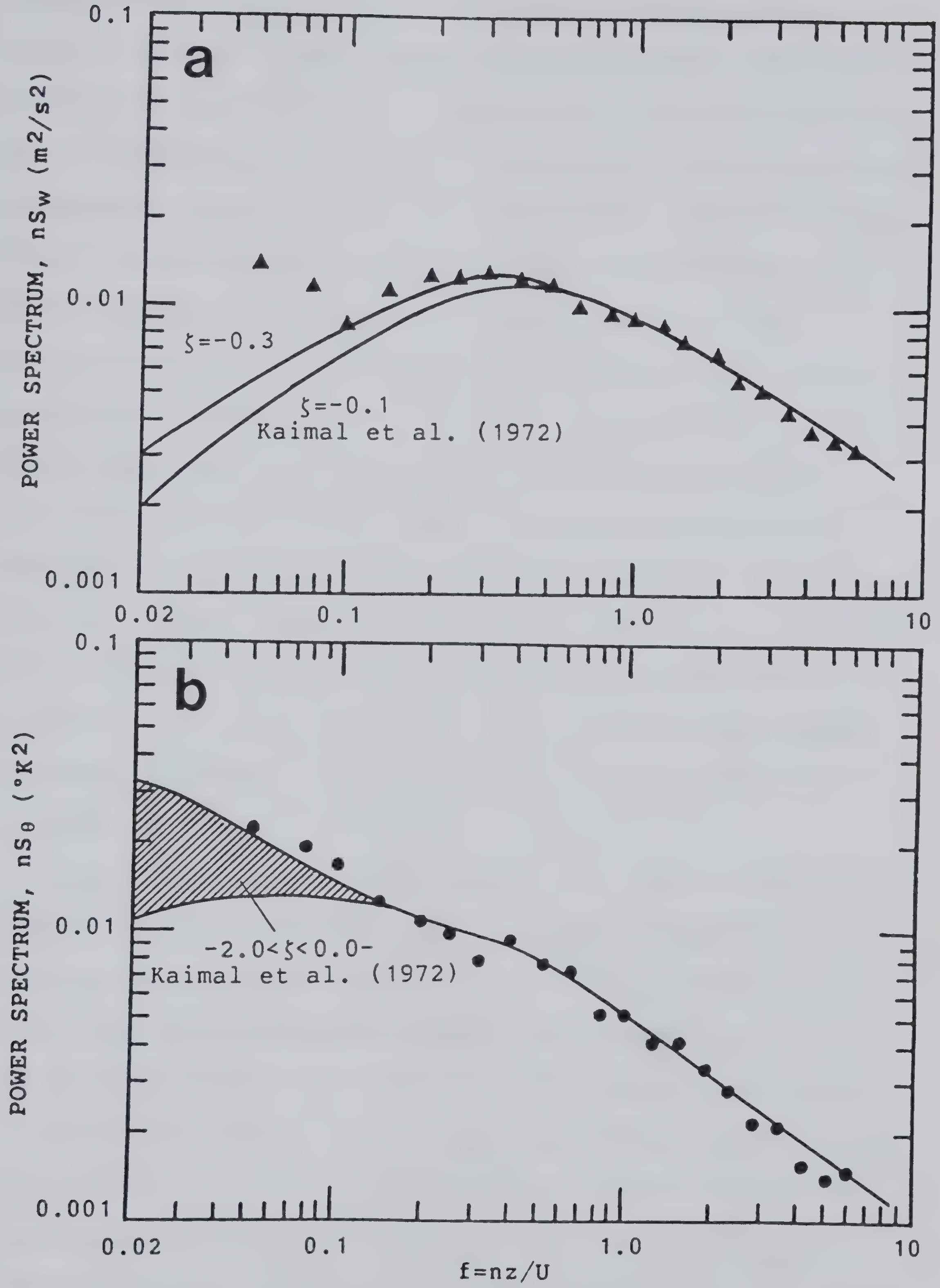


Figure 6.2. Block-averaged sonic w-spectrum (a) and platinum-wire θ-spectrum (b) from 30 minutes of 10 Hz data obtained on October 10, 1982. Solid lines are from Kansas.

temperature, $nS_\theta(n)$, from 30 minutes of turbulence data obtained at the Ellerslie site between 1330 and 1400 local time during an experiment on October 10, 1982. The spectra are the average of 20 modified peridograms, each obtained by transforming time series consisting of 512 points of 10 Hz data. The wind speed at 5 m during the measurements was about 5 m/s, thus making the effect of vertical line averaging negligible. The data were low-pass filtered at 5 Hz; thus, as discussed in Chapter 3, no spectral corrections were applied. For these, and all following measurements, the sonic anemometer was only calibrated relative to a Gill propeller and may, therefore, be in error by ±10%. A more rigorous wind-tunnel calibration is required before the sonic anemometer is used for more quantitative research. Tilt and flow distortions caused by the sensors introduce additional uncertainties which are difficult to quantify.

The solid lines superimposed on the spectra in Figure 6.2 are from the extensive measurements by Kaimal et al. (1972) made at a near ideal site in Kansas. Kaimal et al. normalized their spectra such that all stabilities would converge to a single curve in the inertial subrange. Here, the lines from Kaimal et al. have simply been fitted by eye in the well-defined inertial subrange. Shown in the figure are the Kansas curves for $\zeta = -0.1$ and -0.3. Profile data obtained during the experiment indicated $\zeta \simeq Ri \simeq -0.10$, hence there is fairly good agreement in the shape of the Ellerslie

temperature and vertical wind spectra with those from Kansas. Although not too much faith can be placed in the first couple of low-frequency estimates, it might be noted that a departure of the spectra from the reference curves seems to occur around $f \simeq 0.1$. Given the wind speed of 5 m/s, this corresponds to wavelengths of around 50 m, the distance to the upwind change in surface roughness.

The spectra of vertical wind and temperature, as noted in Chapter 1, can be used to estimate dissipation rates. In terms of the logarithmic spectra, $nS_w(n)$, and reduced wavenumber $f= nz/U$, (1.4) may be rewritten as

$$nS_w(f)= (4/3)\alpha(\epsilon z)^{2/3}(2\pi f)^{2/3}. \tag{6.1}$$

Using $\alpha= 0.55$ from Dyer and Hicks (1982), and choosing $f= 1.5$, which is well into the subrange yet still free of aliasing, gives

$$nS_w(1.5)= 0.164(\epsilon z)^{2/3}, \tag{6.2}$$

or at 5 m,

$$\epsilon= \{2.08 \cdot nS_w(1.5)\}^{3/2}. \tag{6.3}$$

From figure 6.2(a), $nS_w(1.5)=0.0080$ m^2/s^2 which gives $\epsilon=0.0022$ m^2/s^3. For near neutral conditions

$$\epsilon \simeq u_*^3/\kappa z, \tag{6.4}$$

which implies a friction velocity of about 16 cm/s. A profile estimate of $u_* \simeq 20$ cm/s was obtained on this occasion, indicating that the dissipation estimate is quite

reasonable. A covariance estimate can not be given, since horizontal propeller data were not recorded during this experiment. The value of ϵ, unlike a covariance estimate, should not be affected by the low-frequency cut-off imposed by processing short segments of data. Implicit, of course, is the assumption that stationarity existed for a sufficient time prior to the measurements, to allow the inertial subrange to adjust to the present conditions, which was likely the case.

Using β= 0.68, from Dyer and Hicks (1982), and f= 1.5, the dissipation of temperature variance, N, (1.5) becomes

$$nS_\theta(1.5)= 0.152\ N\epsilon^{-1/3}z^{2/3}, \tag{6.5}$$

or at z=5 m,

$$N= 2.25\ \epsilon^{1/3}nS_\theta(1.5). \tag{6.6}$$

From Figure 6.2(b), $nS_\theta(1.5)= 0.0038°K^2$. Using the previous value of ϵ, gives $N= 0.0011°K^2/s$. For near neutral conditions,

$$N\simeq H^2/\{(\rho C_p)^2u_*\kappa z\}, \tag{6.7}$$

thus the subrange value of N corresponds to a heat flux of around 25 W/m^2. This agrees fairly well with a concurrent covariance estimate of 35 W/m^2 obtained by averaging ten blocks, each containing 1536 data values (the standard deviation of this estimate was 12 W/m^2). Again, therefore, a reasonable dissipation rate estimate is obtained from the inertial subrange.

For the October 10 data, stationarity over the 30 minute period considered was probably a good assumption. There are, however, situations when this would not be the case. On September 16, 1982, the prototype data acquisition system was used to collect 15 minute sequences of 10 Hz data at 45 minute intervals from 1700 to 2100 local time. During this evening transitional period, the nature of turbulence undergoes large variations. Figures 6.3 and 6.4 show the vertical wind and the temperature spectra obtained at 1700-1715, 1830-1845, and 2000-2015 which correspond to unstable ($\zeta \simeq -0.1$), near neutral, and stable conditions ($\zeta \simeq +0.1$). The wind remained fairly steady during this period and a single mean value has been used to compute the reduced wavenumber. For the vertical wind spectra there is a slight indictation of a relative shift of energy to higher frequencies with increasing stability, superimposed on the overall drop in the intensity of turbulence.

The changes in the magnitude of the temperature spectra are more dramatic than those in vertical wind. The intensity of the temperature fluctuations drops to a minimum at neutral then increases rapidly as the stable stratification sets in. The breakdown of inertial subrange behaviour in the neutral spectrum, as well as the high frequency end of the unstable spectrum, is probable due to noise problems which are discussed in Section 6.4.

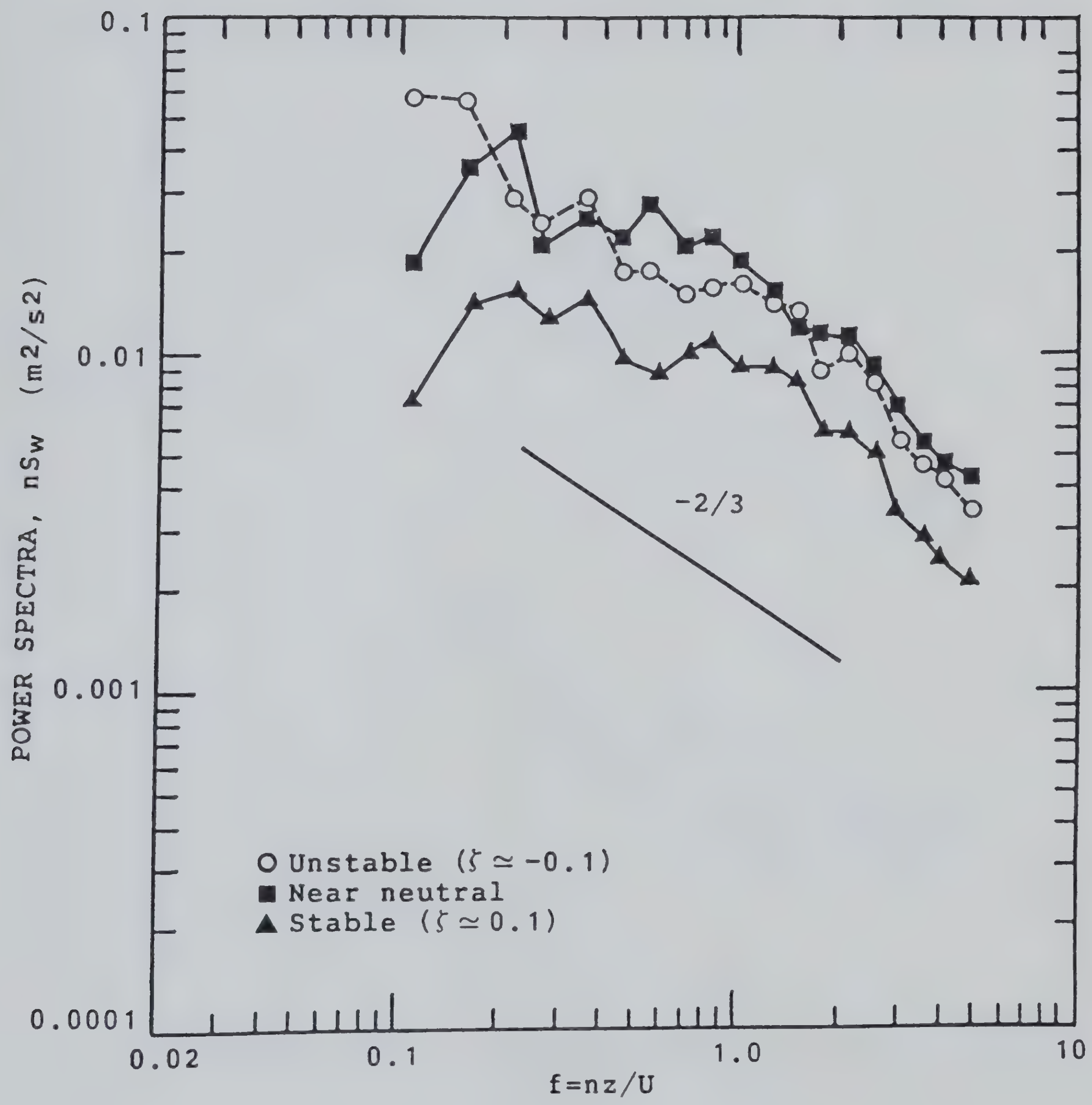


Figure 6.3. Sequence of vertical wind spectra derived from sonic data obtained during sunset at the Ellerslie site on September 16, 1982. Transforms were done on segments of 256 values. Each spectrum is the average of 20 periodograms covering about 15 minutes. Filter corrections have been applied and aliasing corrected for by interpolating the spectra.

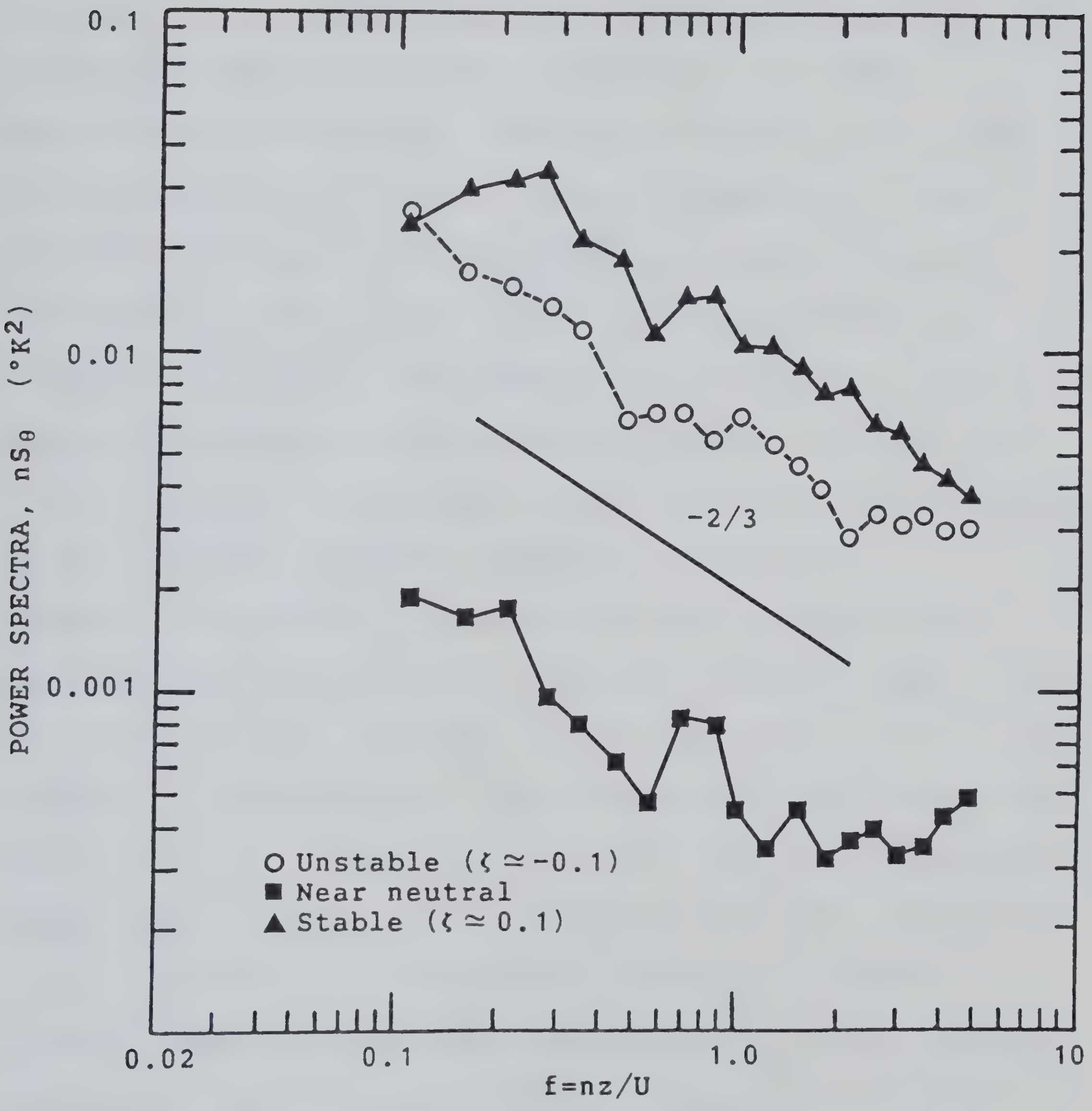


Figure 6.4. Sequence of platinum wire temperature spectra obtained during sunset at the Ellerslie site on September 16, 1982. Spectral computation is the same as in Fig. 6.3.

6.3 Propeller Spectra

The quality of the previous temperature and vertical wind spectra becomes clear upon examining the spectra obtained from propeller anemometers. An experiment on October 9, under near neutral conditions, included measurements of a vertical propeller mounted close to the sonic anemometer. Figure 6.5 shows a comparison of the vertical wind spectra of these two instruments obtained from 30 minutes of 10 Hz data. The failing response of the propeller is clearly seen above $f \simeq 0.3$. Figure 6.6 shows the phase and coherency spectra between the two sensors which, if the response of the sonic is perfect, can be interpreted as the transfer and phase functions of the vertical propeller. The solid lines superimposed on the data are obtained from the horizontal frequency response model given in (2.3) and (2.4) assuming, arbitarily, that $L = 1.5$ m. The comparison is encouraging, with the exception of the phase function at high frequencies. Although the phase estimates become less reliable as the cohereny drops off, the return to zero behaviour of the computed phase is, in part, probably due to the physical separation of the two sensors. The dashed line in Figure 6.6(b) is the modelled phase after making a simple correction, based on Taylor's hypothesis, for a 20 cm along-wind separation.

The response of properly aligned horizontal propellers is expected to be somewhat better than that of a vertical propeller, although no simple method of verifing the

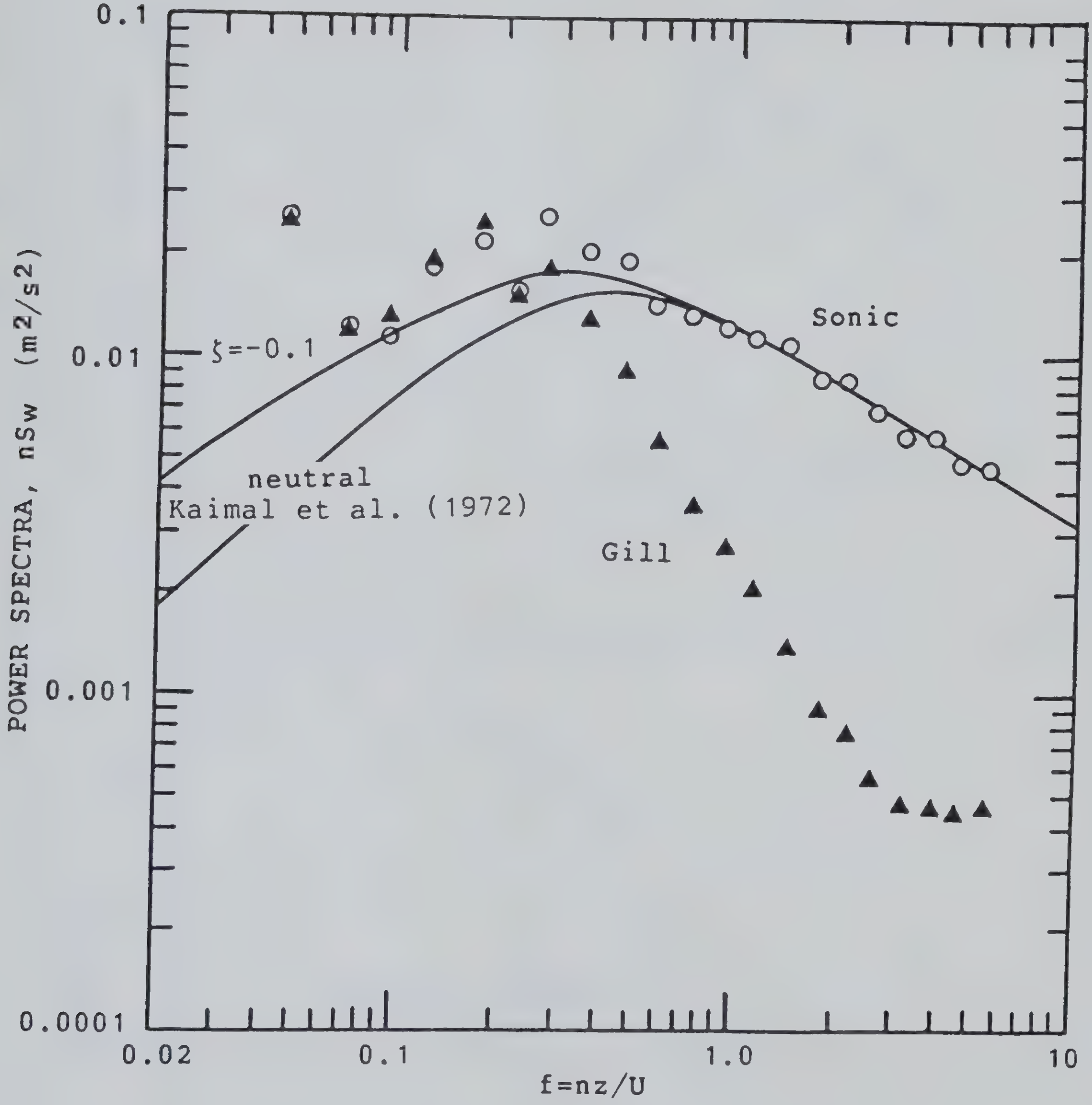


Figure 6.5. Comparison of a Sonic w-spectrum and a vertical Gill spectrum. Spectra are computed from about 30 minutes of 10 Hz data obtained on October 8, 1982. Transforms were on segments of 512 values. No spectral corrections have been applied.

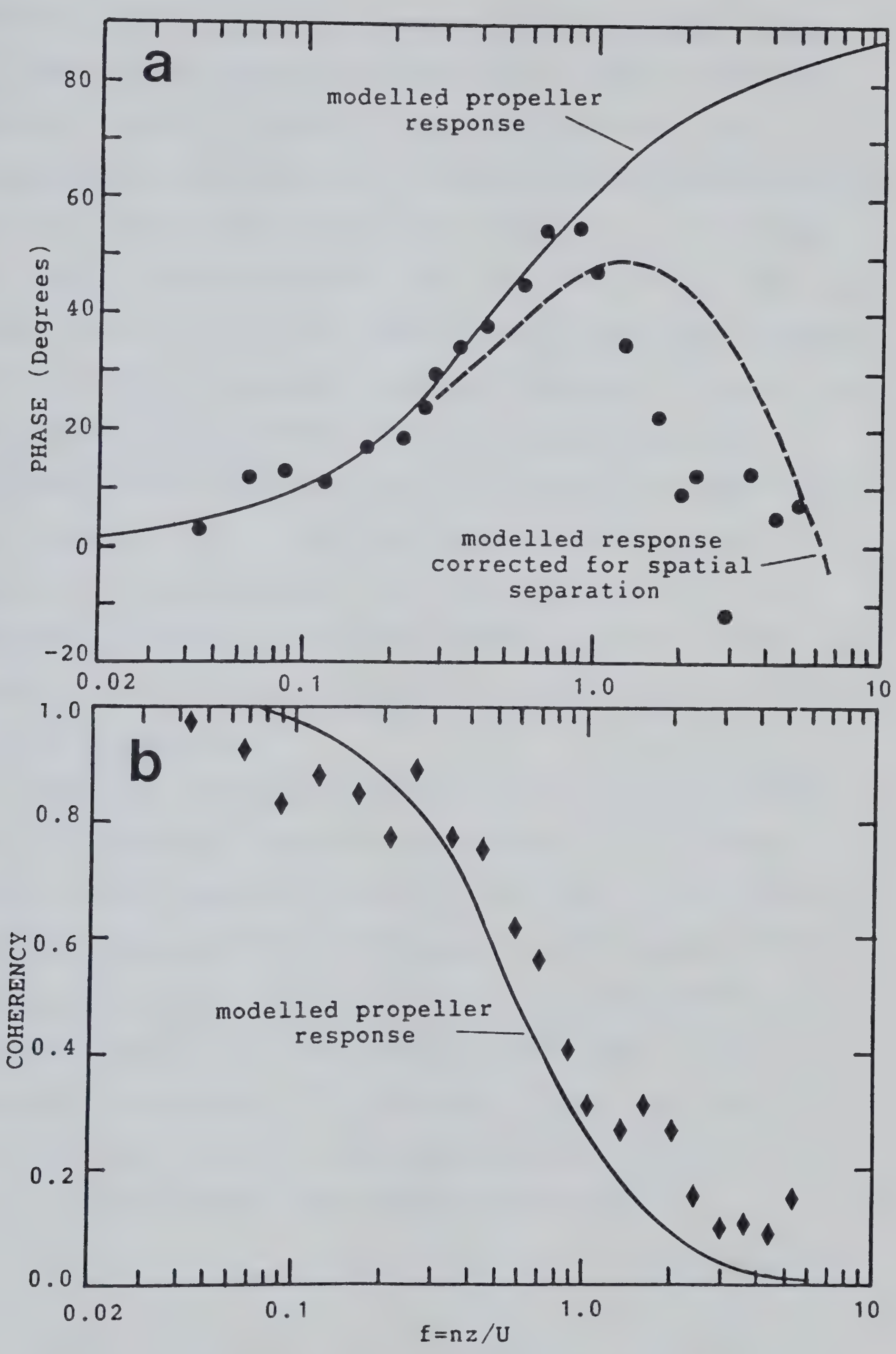


Figure 6.6. The phase (a) and coherency (b) between the sonic anemometer and a vertical Gill propeller.

transfer function given in Chapter 2 was available. Figure 6.7 shows the u-spectra for 30 minutes of data concurrent with the vertical wind data used in Figure 6.6. The longitudinal component was obtained by cosine correcting the data from the orthogonal pair of propellers, one of which was closely aligned with the mean wind. The solid lines from Kaimal et al. (1972) have been fitted in the region around f= 0.2 where the propeller response correction is small, yet the statistical confidence is still fairly high. The placement of the line in Figure 6.7 implies, by comparison with Kaimal et al. (1972), a value of u_* of about 20 cm/s, which is in good agreement with concurrent profile and covariance estimates obtained on this occasion. As expected, however, the uncorrected spectrum falls off much too quickly at high frequencies.

When the horizontal propeller spectrum is corrected, the high frequency end shoots up sharply in an aliased fashion. This apparent aliasing, also seen in the vertical propeller spectrum shown in Figure 6.6, was typical of all computed propeller spectra. However, the sharp high frequency cutoff of these sensors should result in a reduction of aliasing. The observed effect, therefore, is probably due to noise rather than aliasing. The source of this noise is probably the digitizing process.

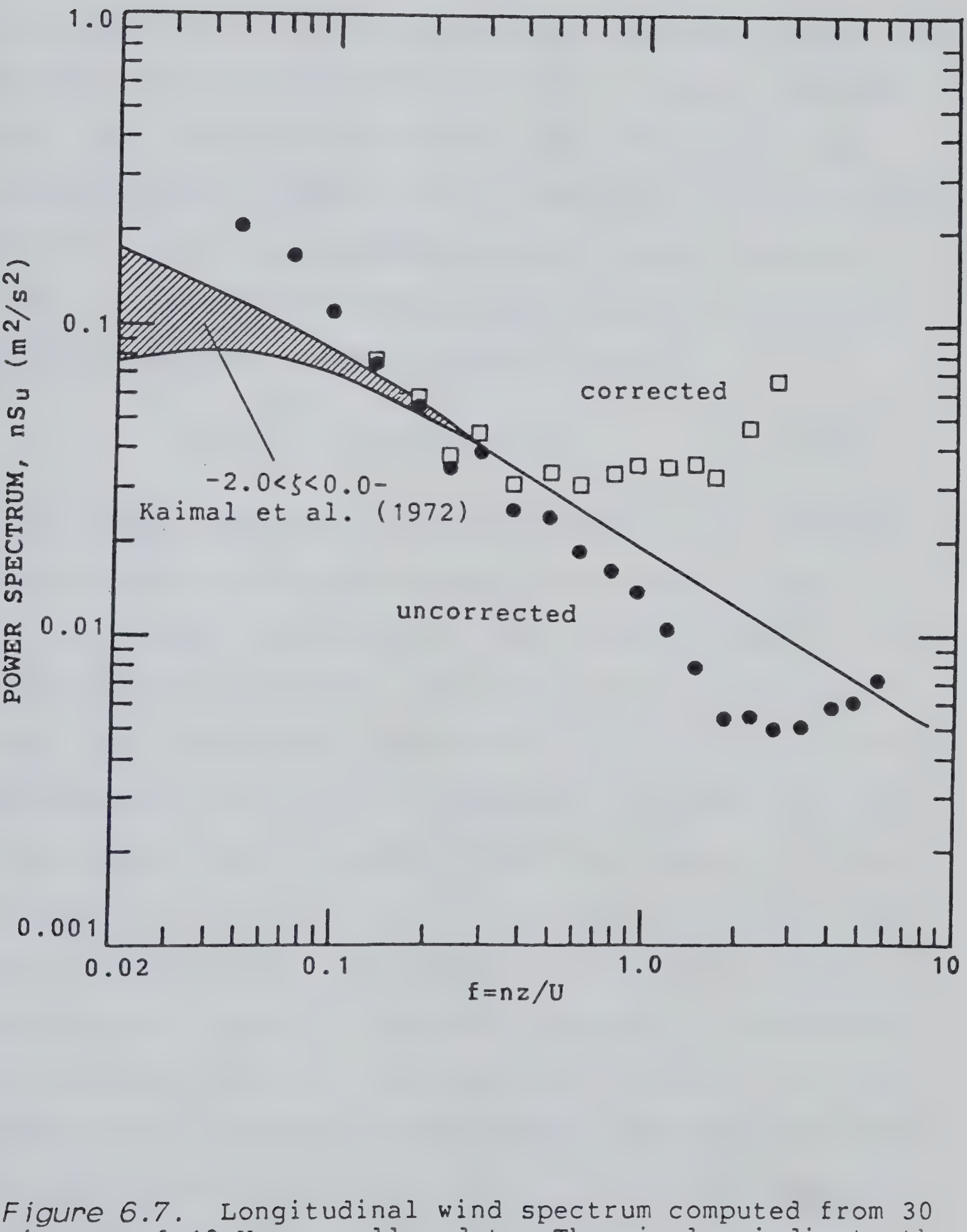


Figure 6.7. Longitudinal wind spectrum computed from 30 minutes of 10 Hz propeller data. The circles indicate the spectrum computed from cosine-corrected data. The squares indicate the spectrum after attempting to correct for propeller frequency response using L= 1.1 m.

6.4 Noise due to ADC Resolution

The validity of using an 8-bit ADC depends upon the severity of resolution related problems. The error introduced by the digitizing process is bounded between ±R/2 where R is the resolution of the ADC. It seems reasonable to assume that, for turbulence data, any value within the allowable range is equally likely and the mean will be zero. This corresponds to a probability distribution function which is a constant 1/R between ±R/2 and zero everywhere else. The variance of such a distribution is

$$\sigma_n{}^2 = \int_{-R/2}^{+R/2} (1/R)x^2dx = R^2/12 \qquad (6.8)$$

If the actual turbulence data are statistically independent of the error introduced by digitizing and this noise is white then there is expected to be a contribution of $R^2/12n_0$ to each spectral estimate. By reducing the resolution of actual data to simulate the effects of an A-to-D convertor, this has been shown to be an adequate description of the noise problem. As an example, Figure 6.8 shows a 15 minute vertical wind spectrum computed from data which had their resolution numerically reduced from 0.80 cm/s (8 bits) to 13 cm/s (4 bits). Despite this large reduction in resolution, the effect on the overall variance is only about 10%. The solid line on Figure 6.8 shows the ADC error expected on the basis of the previous argument and it is in good agreement with the computed spectrum.

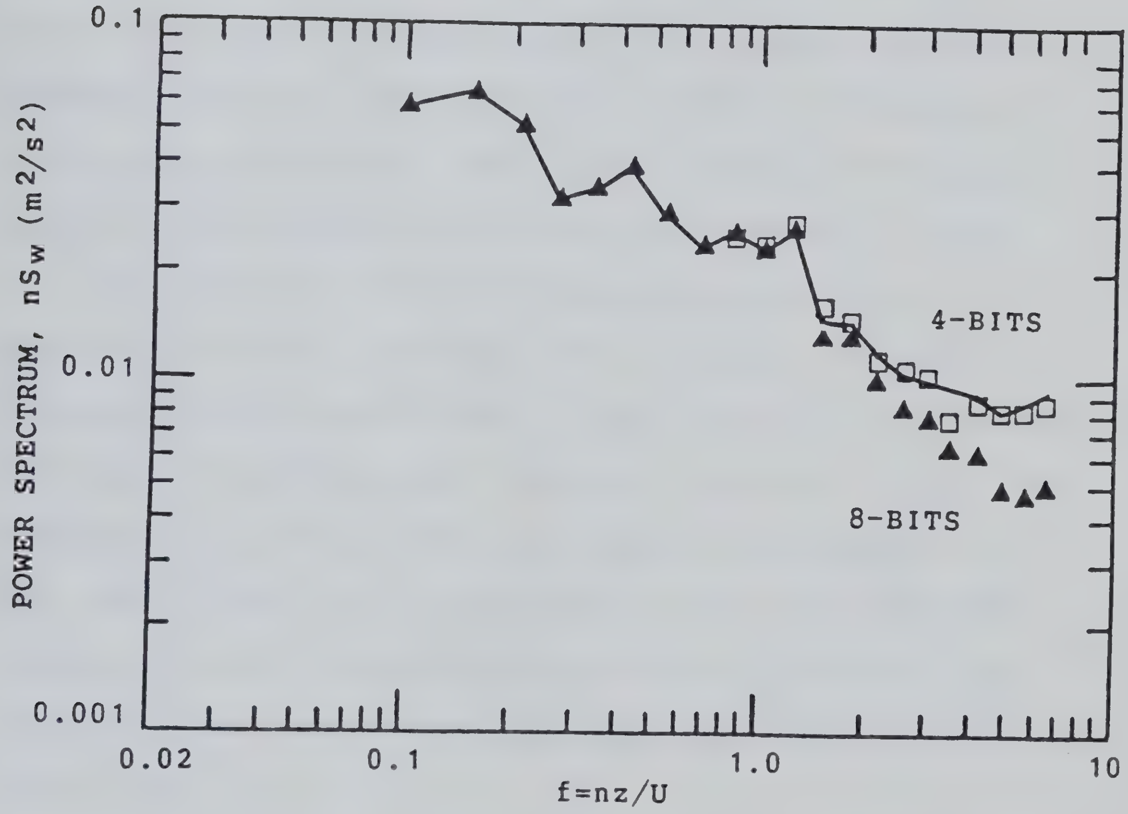


Figure 6.8. Effect of ADC resolution on a typical power atmospheric power spectrum. The triangles indicate a vertical wind spectrum computed from a set of 8-bit data which makes good use of the full range of the ADC. The squares indicate the spectrum of the same set of data computed after reducing the resolution of the data to 4 bits. The solid line indicates the expected effect of reduced resolution assuming it acts to introduce white noise.

The resolution problem has affected the propeller data most severely because relatively low gain settings were customarily used in order to accomodate potentially large variations which could be caused by wind shifts. Although high-pass filtering reduces this problem, it was avoided because of the need to reconstruct the absolute signal levels in order to perform cosine corrections. Of course, improper use of the gain settings can make resolution a problem for any channel. The neutral temperature spectrum in Figure 6.4 is probably suffering from resolution noise.

The ADC noise on any two channels should be uncorrelated, thus covariance and cospectra calculations should be fairly insensitive to ADC resolution. Following the approach used for power spectra, Figure 6.9 shows the cospectra of vertical wind and temperature obtained from data with 8 and 4 bits of resolution. The two cospectra agree very well. The differences which do occur at high frequencies are associated with amplitudes well below the noise level.

6.5 Cospectra

Cospectra reveal the relative importance of different scales of motion to the total flux and are thus of considerable interest. Similarity theory predicts an $n^{-7/3}$ behaviour in the inertial subrange of both $C_{w\theta}$ and C_{wu}, which should show up as a -4/3 slope on the usual logarithmic plot. Figure 6.10 shows $nC_{w\theta}$ corresponding to

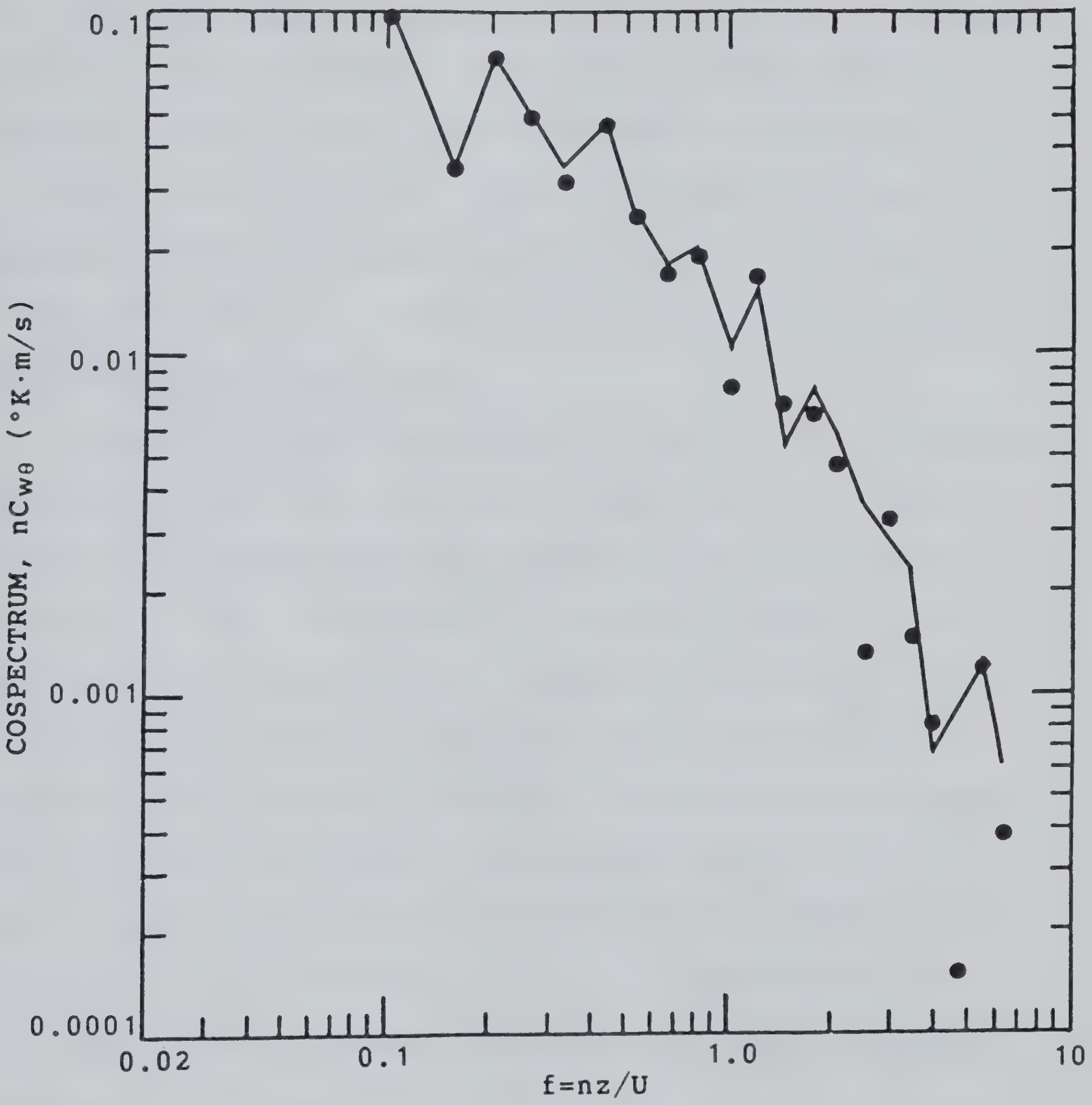


Figure 6.9. Effect of ADC resolution on a typical atmospheric cospectrum. The line indicates a cospectrum $nC_{w\theta}$ computed from 8-bit data. The circles are the cospectral estimates after reducing the resolution to 4 bits.

the earlier sonic and platinum wire power spectra for October 10 (Figure 6.2). The cospectra exhibit more scatter than the corresponding power spectra. This is indicative of the need for longer averaging times for cospectra (fluxes) relative to power spectra (variances) as discussed by Wyngaard (1973). In the inertial subrange of Figure 6.2, $nC_{w\theta}$ seems to fall off just slightly more rapidly than expected. Since the area beneath this curve is proportional to the heat flux, the resulting loss of flux at the low frequencies is clearly seen.

It would be nice to be able to show a graph of momentum cospectra of the same quality as Figure 6.10. Figure 6.11 shows nC_{wu} obtained from the October 9 data. It is not even possible to plot these data in the usual manner, since numerous sign changes occur; instead, a semi-logarithmic graph of nC_{wu} is given. While the use of horizontal propellers is expected to degrade the momentum cospectrum, site and actual atmospheric anomolies have probably contributed to the poorly behaved result in Figure 6.11. Stationarity was expected to be quite good and this is confirmed by the agreement between the 15 and 30 minute cospectra in Figure 6.11.

Somewhat more encouraging are the nC_{wu} graphs shown in Figure 6.12. These three cospectra were obtained from the September 16 data corresponding to the power spectra considered in Section 6.2. They have been corrected for differences in propeller and sonic response as discussed in

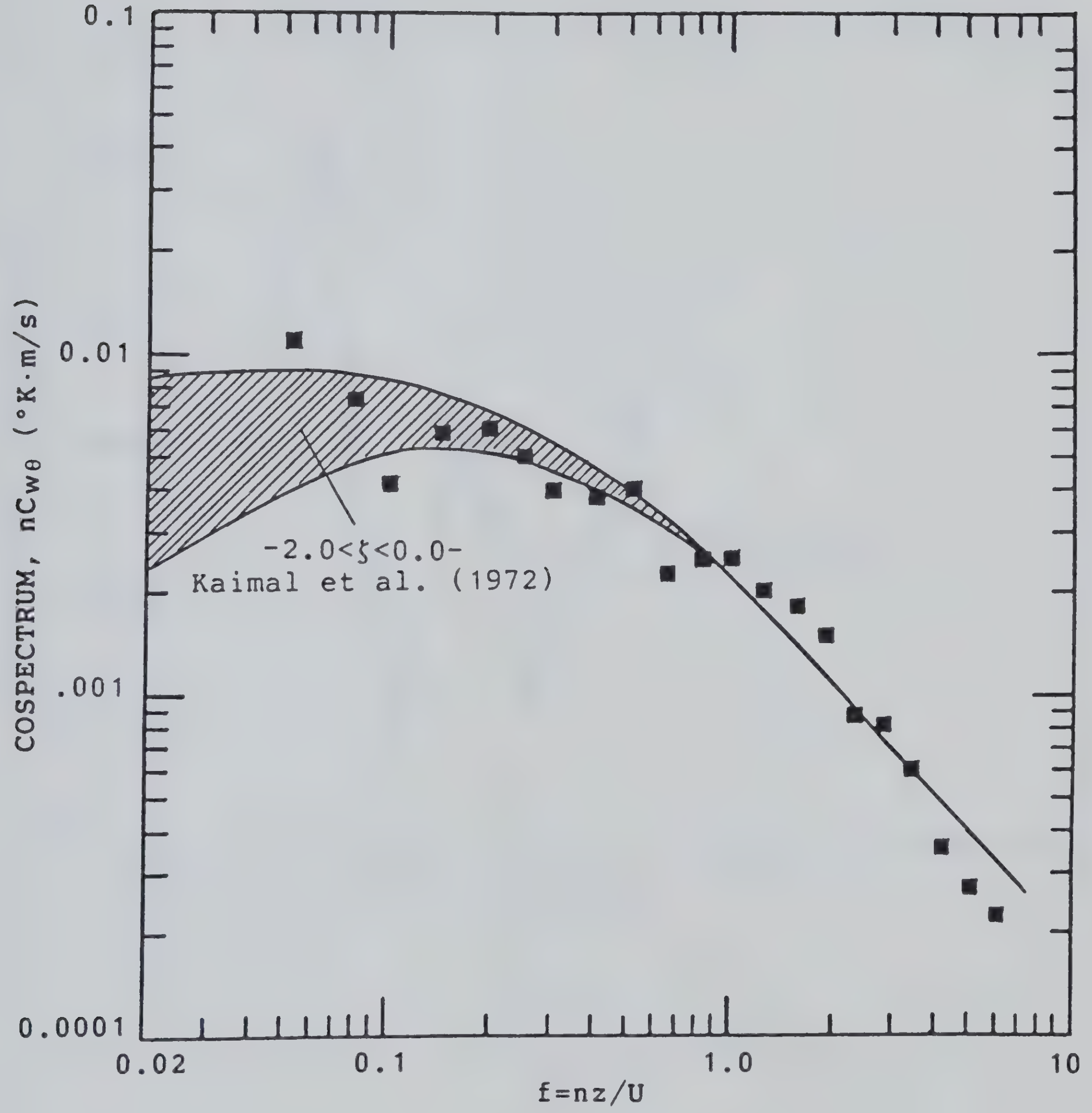


Figure 6.10. A cospectrum of vertical wind and temperature computed from 30 minutes of sonic and platinum wire data corresponding to Fig. 6.2. The solid lines are from Kaimal et al. (1972).

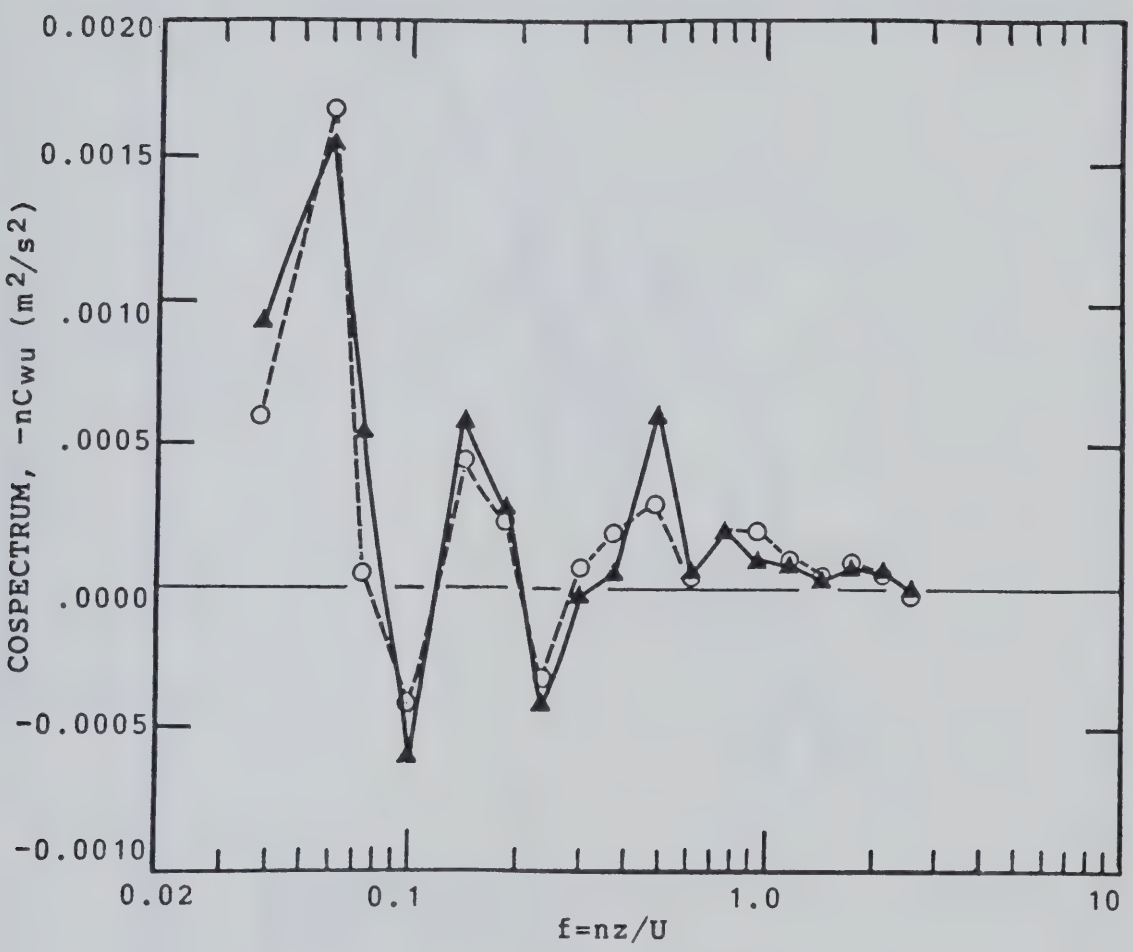


Figure 6.11. Momentum cospectra, $-nC_{wu}$, from 10 Hz data obtained on October 9, 1982. The solid line corresponds to 30 minutes of data while the dashed line was obtained using only using the first 15 minutes. No spectral corrections were applied.

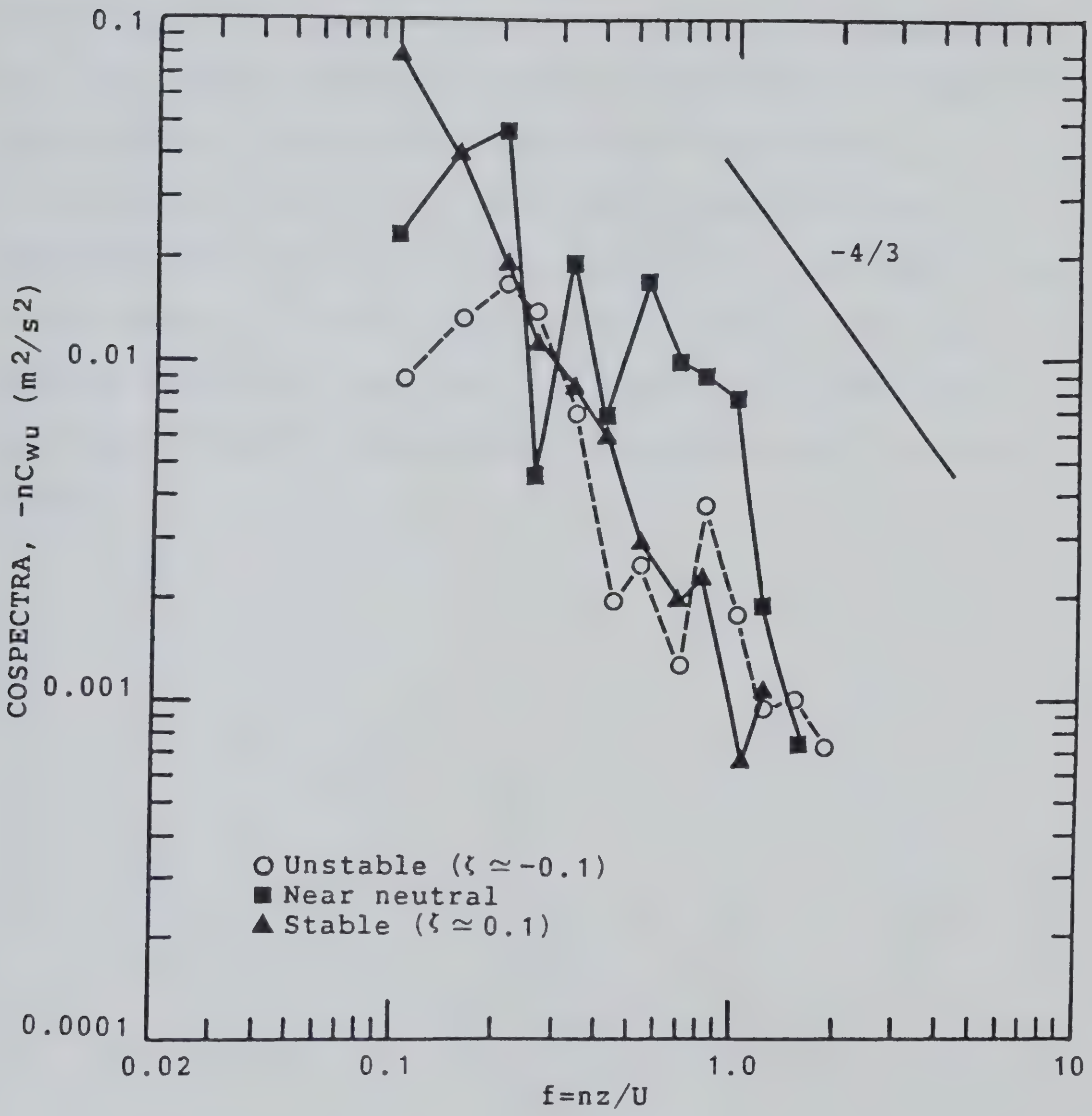


Figure 6.12. Sequence of momentum cospectra, $-nC_{wu}$, from 15 minute segments of 10 Hz data obtained on September 12, 1982. Corrections for propeller response, as discussed in Chapter 2, were applied.

Chapter 2. At higher frequencies than those shown, the spectra change sign and, therefore, were dropped. The scatter of the near neutral nC_{wu} is large, which is not surprising since stationarity is a rather poor assumption for this period. Overall, however,these spectra do seem to be consistent with a -4/3 slope in the range shown.

Finally, Figure 6.13 shows the sequence of $nC_{w\theta}$ for September 16. For the unstable case, it is necessary to plot $-nC_{w\theta}$. Again, a lot of scatter can be seen in the near neutral cospectrum, but the -4/3 slope of the inertial subrange is easily identified in the stable and unstable cases.

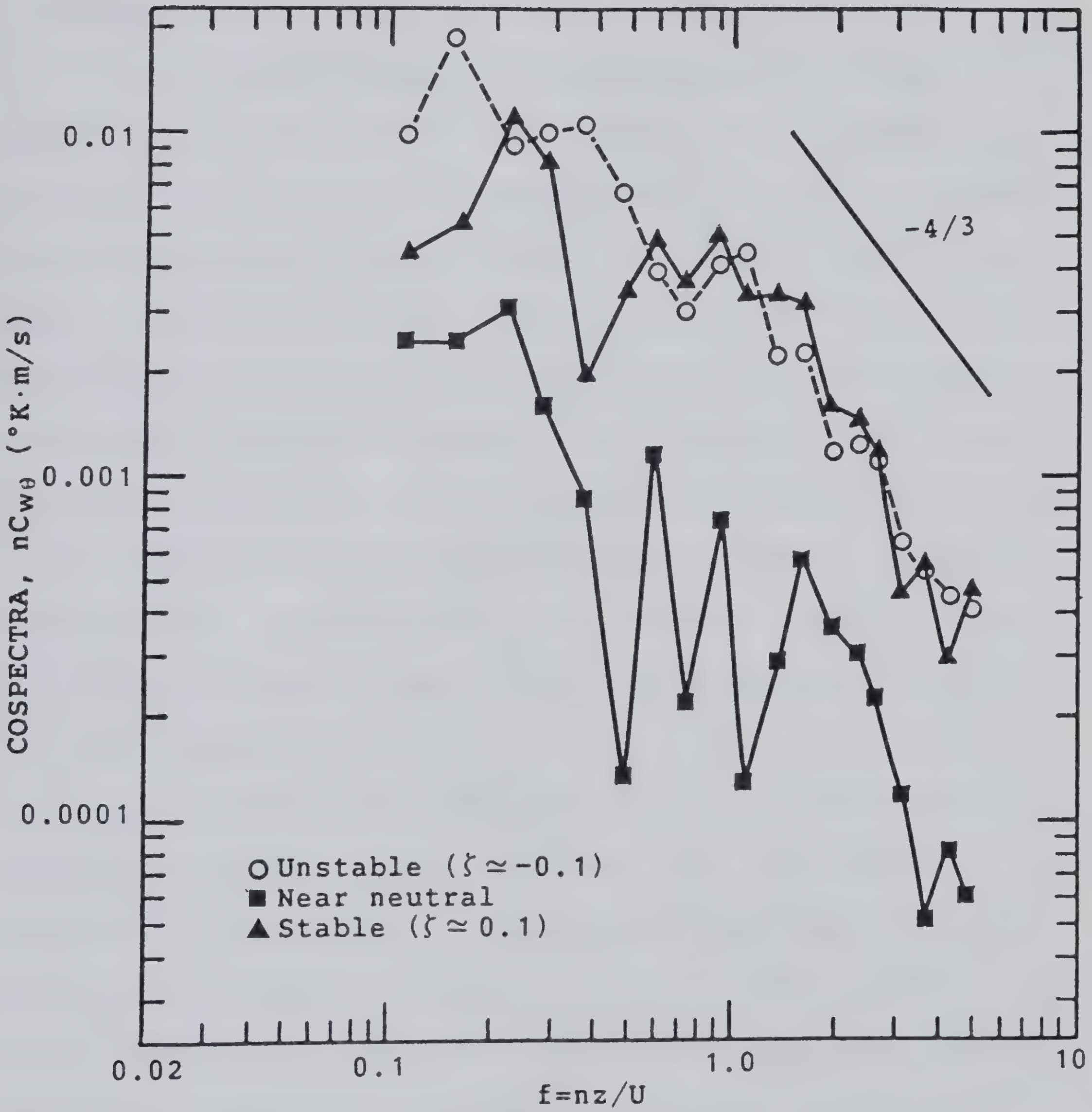


Figure 6.13. Sequence of heat cospectra, $nC_{w\theta}$, from 15 minute segments of 10 Hz data obtained on September 12, 1982. In the stable case, $-nC_{w\theta}$ is plotted.

Chapter 7

SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

7.1 Summary

A microcomputer-based, micrometeorological data acquisition system has been assembled and successfully used to collect and transfer turbulence data to a host computer for processing. The system is small, portable (powered by common 12 V lantern batteries) and relatively inexpensive. Meteorlogical instrumentation consists of a sonic anemometer for vertical wind measurements, an orthogonal pair of Gill propeller anemometers for horizontal wind measurements, and a thin platinum wire (25 μm diameter) to measure temperature fluctuations. A standard Gill UVW array was used to mount all four instruments. Cost of the resulting wind-temperature array was about $2500.

Analog conditioning, multiplexing, and 8-bit analog-to-digital circuitry which interface the instruments to an RCA VIP-711 single-board microcomputer have been assembled and described. The microcomputer controls data collection at 5, 10, 15 or 20 Hz (same for all channels). Simultaneous sampling is approximated by fast sequential digitizing (1.25 ms per channel). There is enough on-board memory to buffer up to 3328 data bytes. When this available space is filled, data collection is stopped and the data are transfered to standard audio cassette tape at 100 bytes per second. The possiblity of achieving continuous data collection at up to

100 bytes per second has been considered and appears feasible with improved software. The microcomputer also controls the retrieval and transmission of the data to a host computer via an RS-232C interface. This process involves reading data from tape, converting them from binary to ASCII code, and then transmitting the ASCII bytes in serial form at 600 baud. This interface was implemented predominantly in software. The only hardware required was a simple RS-232C driver. Cost of the computer, tape-recorder, and electronic components required for this project came to around $500. Documented assembly and machine language listings of the prototype software are provided.

A series of experiments to test the prototype system were undertaken at the University of Alberta Research Farm (Ellerslie) located approximately 10 km south of downtown Edmonton, Alberta. Statistical and spectral analysis was performed on many of the data sets obtained. The results of a few selected cases were presented. The -5/3 and -7/3 behaviours expected in the inertial subrange of atmospheric power spectra and cospectra, respectively, were generally found to be followed fairly closely. Reasonable estimates of dissipation rates were obtained from the subrange of sonic vertical wind and platinum-wire temperature spectra.

7.2 Conclusions

This work has demonstrated that it is currently possible to construct a microcomputer-based data acquisition system suited to the study of atmospheric turbulence for around $500 (not including the cost of meteorological instrumentation). This is at least an order of magnitude less than any comparable commercial systems currently available. Hardware assembly is relatively easy and neither specialized development tools nor a great deal of technical expertise are required.

The RCA COSMAC VIP-711 training microcomputer was found to be well suited to this project, operating reliably during two months of intermittent field testing. The commercial quality electronic components of this and other comparable microcomputers does, however, restrict exposed operation to temperatures above 0°C.

The analysis of data collected with the prototype system has confirmed that the low-cost, CA27 sonic anemometer and a very simple platinum wire thermometer can be used to study the spectra of turbulence at frequencies up to at least 10 Hz and to obtain reasonable subrange estimates of dissipation rates. Propeller anemometers, on the other hand, provide little information above about 1 Hz. Data analysis also indicates that 8-bit analog-to-digital conversion can give sufficient resolution for the study of turbulence provided that analog amplification and offsets are used judiciously.

7.3 Improvements

Being microcomputer-based, the prototype system has a great deal of potential for further development. The following are a few suggestions for improvements:

1. As discussed in Chapter 4, it should be possible to provide continuous data collection at 100 bytes per second by modifying just the software. This is the most important improvement which needs to be made to the current system.
2. The software could also be improved to permit different channels to be sampled at different frequencies. In particular, the propeller anemometers could be sampled at half the frequency of the sonic anemometer and platinum wire thermometer.
3. The data collection and RS-232C interface software could be transfered to Read-Only-Memory. The VIP-565 EPROM programmer and VIP-560 EPROM interface board are available to facilitate this operation. Placing the programs in EPROM would simplify field operation, although it should be noted that loading programs from tape caused no difficulties during the initial testing.
4. Conversion to 12-bit data words would provide sufficient resolution to eliminate the need for user adjustable gain and offset controls, thus significantly simplifing field operation of the system. However, cost of a sufficiently fast 12-bit ADC is many times greater than the 8-bit unit used here. In addition, interfacing a

12-bit ADC will be more difficult since the extra 4 bits will have to be multiplexed onto the 8-bit input port. Using 2 bytes per sample will allow writing only 50 data values per second to tape. This should be sufficient, however, if continuous data collection is implemented.

5. The present 600 baud transfer to the host computer is rather slow. Transfer rates of 1200 or 2400 baud could probably be implemented in software although these values would not be particularily useful since they are not compatible with the available terminal rates. A 9600 baud interface which would allow the system to tie in with an available CRT terminal would be very convenient although this could not be done in software alone; a UART chip, baud rate generator, and accompanying circuitry would be required.
6. The inclusion of real-time output, notably covariances, would be useful in pointing out instrument malfunctions or unusual atmospheric conditions. There is certainly sufficent time for the microprocessor to do some real-time calculations; in the present software most of its time is spent waiting for an A-to-D conversion to start or finish. Using this spare time, however, would require implementing much more elaborate interrupt-based software. In addition, since a video monitor is not well suited to field use, a device to provide real-time output, preferable a small printer, would also have to be purchased and interfaced to the microcomputer.

Implementing continous data collection, as discussed in Chapter 4, and simultaneous real-time output would present a formidable task probably best left to an experienced microcomputer system designer.

7.4 Future Considerations

While the COSMAC VIP-711 microcomputer seems to have been a good choice for this project, RCA has now stopped manufacturing it as noted in Chapter 4. In retrospect this is not too surprising. The VIP system was originally introduced in 1978; five years is a long time in the microcomputer business. There are, however, other training microcomputers based on the RCA 1802 processor which could be used for this project. The ELF from Neutronics and the VENTURE from Quest are two likely candidates. For anyone already familar with (or willing to learn) a different microprocessor language, there is no need to restrict the choice to a 1802-based system. Aside from low-cost, a tape cassette interface running at 1000 or more baud and a sufficient number of decoded input/output lines are the main features to look for when selecting a microcomputer. The $200 MPF-1 from Multitech, for example, appears to be a promising Z80-based microcomputer, but its cassette interface operates at only 16.5 bytes per second, which makes it unsuitable for the present application.

There seems to be little hope of seeing reliable interfaces for audio cassette recorders at much over 1000

baud. Hard disc or bubble memory systems with over a megabyte of non-volatile storage have potential, but, at present, are still very expensive. Thus, cassette tape is likely to remain the cheapest answer to relatively slow, sequential data collection for some time to come.

What about the new generation of 16-bit microcomputers? At present, comparable 16-bit single board systems are still quite expensive. Before long, however, such systems will become widely available and greatly facilitate the use of 12-bit A-to-D converters in a project such as this. In the meanwhile, there is much that can be accomplished with the inexpensive 8-bit systems currently available.

REFERENCES

Barat, J., 1982: A high resolution ionic anemometer for boundary layer studies. *J. Appl. Meteor.*, **21**, 1480-1488.

Bingham, C., M. D. Godfrey and J. W. Tukey, 1967: Modern techniques of power spectra estimation.*IEEE Trans. of Audio and Electroacoustics*, **AU-15**, 58-66.

Brooks, R. R., 1977: Effective dynamic response of paired Gill anemometers. *Bound. Layer. Meteor.*, **11**, 33-37.

Campbell, G. S., and M. H. Unsworth, 1979: An inexpensive sonic anemometer for eddy correlation. *J. Appl. Meteor.*, **18**, 1072-1077.

Drinkrow, R., 1972: A solution to the paired Gill anemometer response function. *J. Appl. Meteor.*, **11**, 76-80.

Dyer, A. J. and B. B. Hicks, 1982: Kolmogoroff constants at the 1976 ITCE. *Bound. Layer. Meteor.*, **22**, 137-150.

Dyer, A. J., B. B. Hicks and K. M. King, 1967: The Fluxatron-a revised approach to the measurement of eddy fluxes in the lower atmosphere. *J. Appl. Meteor.*, **6**, 408-413.

Fichtl, G. H., and P. Kumar, 1974: The response of a propeller anemometer to turbulent flow with the mean wind vector perpendicular to the axis of rotation. *Bound. Layer Meteor.*, **6**, 363-379.

Hicks, B. B., 1972: Propeller anemometers as sensors of atmospheric turbulence. *Bound. Layer Meteor.*, **8**, 255-259.

Hogstrom, U., 1982: A critical evaluation of the aerodynamical error of a turbulence instrument. *J. Appl. Meteor.*, **12**, 1838-1844.

Horst, T. W., 1973a: Corrections for response errors in a three-component propeller anemometer. *J. Appl. Meteor.*, **12**, 716-725.

Horst, T. W., 1973b: Spectral transfer function of a three-component sonic anemometer. *J. Appl. Meteor.*, **12**,1072-1075.

Kaimal, J. C., 1968: The effect of vertical line averaging on the spectra of temperature and heat flux. *Quart. J. Roy. Meteor. Soc.*, **94**, 149-155.

Kaimal, J. C., J. C. Wyngaard, Y. Izumi and O. R. Cote, 1972: Spectral characteristics of surface-layer turbulence. *Quart. J. Roy. Meteor. Soc.* , **98**, 563-589.

Kaimal, J. C., J. C. Wyngaard and D. A. Haugen, 1968: Deriving power spectra from a three-component sonic anemometer. *J. Appl. Meteor.*, **7**, 827-837.

Larsen, S. E., F. W. Weller and J. A. Businger, 1979: A phase-locked loop continuous wave sonic anemometer thermometer. *J. Appl. Meteor.*, **18**, 562-568.

McBean, G. A., 1972: Instrument requirements for eddy correlation measurements. *J. Appl. Meteor.*, **11**, 1078-1084.

Mitsuta, Y., 1966: Sonic anemometer thermometer for general use. *J. Meteor. Soc. Japan*, **44**, 12-24.

Peatman, J. B., 1972: *Microcomputer-Based Design.* McGraw Hill, 540 pp.

Redford, T. G., Jr., S. B. Verma and N. J. Rosenberg, 1981: Drag anemometer measurements of turbulence over a vegetated surface. *J. Appl. Meteor.*, **20**, 1222-1230.

Shuttleworth, W. J., D. D. McNeil, and C. J. Moore, 1982: A switched continuous-wave sonic anemometer for measuring surface heat fluxes. *Bound. Layer Meteor.*, **23**, 425-448.

Smith, S. D., 1980: Evaluation of the Mark 8 thrust anemometer for measurements of boundary layer turbulence. *Bound. Layer Meteor.*, **19**, 273-291.

Weihofen, U. and R. Woehl, 1981: A low-cost, multipurpose data acquisition device based on a microprocessor. *Agric. Meteor.*, **24**, 111-116.

Wieringa, J., 1972: Tilt errors and precipitation effects in Trivane measurements of turbulence fluxes over open water. *Bound. Layer Meteor.*, **2**, 406-426.

Wyngaard, J. C., J. A. Businger, J. C. Kaimal, S. E. Larsen, 1982: Comments on "A revaluation of the Kansas mast influence on measurements of stress and cup anemometer overspeeding". *Bound. Layer Meteor.*, **22**, 245-250.

Wyngaard, J. C., 1973: On Surface-Layer Turbulence, *Workshop on Micrometeorology*, Amer. Meteor. Soc. Publication, 101-148.

APPENDIX

This appendix contains one version of the machine language programs used by the prototype data acquisition system. The data collection software permits the user to input the number of channels, n, to be sampled, and then uses the first n channels selected sequentially from W, T, U, V, X, and Y. If this order is not satisfactory for a particular experiment it can be easily modified by altering the initializations in locations #0032 through #0055.

Both data collection and Host interfacing software use routines in the RCA COSMAC VIP-711 operating system. These routines have been disassembled and documented and are included here for completeness. Except for the RCA subroutines, the programs are loaded from cassette tape into RAM locations #0000 to #01FF. Only one program resides in RAM at any one time.

The programs have been developed with the aid of a version of the RCA COSMAC single pass cross assembler provided by the University of Alberta Computing Services in the public file *COSMAC. Certain assembler language commands are missing in this limited implementation and machine language hexcodes must be used in their place. Most notable of these are the REQ (#7A) and SEQ (#7B) commands which control the Q line.

```
LOC  COSMAC CODE  SOURCE LINE
0000              ..-------------------------------------
0000              ..COSMAC DATA COLLECTION SOFTWARE
0000              ..WRITTEN BY PAUL HOPPS (1982)
0000              ..-------------------------------------
0000              ..REGISTER DESIGNATIONS:
0000              ..R1   Pointer to I/O Control Bytes
0000              ..R2.0 Channel Counter
0000              ..R2   Address of Key-Input
0000              ..R3   RCA Tape-Write Program Counter
0000              ..R4   Main Program Counter
0000              ..R6   Pointer to next buffer location
0000              ..R7   Keypad-Input subroutine P counter
0000              ..RA.0 Number of pages per channel
0000              ..RA.1 Number of channels
0000              ..RB   Total number of pages in buffer
0000              ..RC   RCA Write-a-Bit P Counter
0000              ..RE.0 Page counter for RCA routine
0000              ..RF   Keypad debounce counter
0000              ..RF   Remaining buffer space counter
0000              ..-------------------------------------
0000 6380         ,#6380    ..Turn cassette power off
0002 90           GHI  0    ..Initializations, make R4
0003 B4           PHI  4    ..the main program counter.
0004 B7           PHI  7
0005 F809         LDI  #09
0007 A4           PLO  4
0008 D4           SEP  4
0009 F8A1A7       #A1->R7.0 ..Call Key-Input subroutine
000C D7           SEP  7    ..to input the number of
000D BA           PHI  A    ..channels (Store in RA.1).
000E FF07         SMI  #07  ..Check to make sure the
0010 3B15         BL OKCHN  ..number of channels is <7.
0012 7B           ,#7B      ..If no, turn on Q tone
0013 3013   ENDCH:BR ENDCH  ..and terminate in a loop.
0015 D7     OKCHN:SEP  7    ..Else input pages per
0016 AA           PLO  A    ..channel (store in RA.0).
0017 9A           GHI  A    ..Get number of channels
0018 AF           PLO  F    ..(store in RF.0)
0019 94           GHI  4
001A AB           PLO  B
001B 8B      PLUS:GLO  B    ..Compute total number of
001C F4           ADD       ..pages required by
001D AB           PLO  B    ..multiple additions (X=2)
001E 2F           DEC  F    ..since the COSMAC does
001F 8F           GLO  F    ..not have a multiply
0020 3A1B         BNZ PLUS  ..instruction (store
0022 8B           GLO  B    ..result in RB.0).
0023 FD0A         SDI  #0A  ..Is there
0025 332B         BGE OKRAM ..enough memory?
0027 7B           ,#7B      ..If no, turn on Q tone
0028 3028   ENDRM:BR ENDRM  ..and terminate in this loop
002A D7           SEP  7    ..else wait for clock to start.
002B F881BC OKRAM:#81->RC.1
```

```
LOC  COSMAC CODE  SOURCE LINE
002E F8FBA1       #FB->R1.0
0031 E1           SEX 1
0032 F8A5         LDI #A5   ..Multiplexer Initializations:
0034 73           STXD
0035 F885         LDI #85   ..00F0  #80
0037 73           STXD      ..00F1  #A0
0038 F8A4         LDI #A4   ..00F2  #81
003A 73           STXD      ..00F3  #A1
003B F884         LDI #84   ..00F4  #82
003D 73           STXD      ..00F5  #A2
003E F8A3         LDI #A3   ..00F6  #83
0040 73           STXD      ..00F7  #A3
0041 F883         LDI #83   ..00F8  #84
0043 73           STXD      ..00F9  #A4
0044 F8A2         LDI #A2   ..00FA  #85
0046 73           STXD      ..00FB  #A5
0047 F882         LDI #82
0049 73           STXD      ..In the I/O control bytes,
004A F8A1         LDI #A1   ..an '8' is used to select
004C 73           STXD      ..a MUX channel and an 'A'
004D F881         LDI #81   ..is used to initiate an
004F 73           STXD      ..A/D conversion. In this
0050 F8A0         LDI #A0   ..program channels 0,1,2,3,4,
0052 73           STXD      ..and 5 (W,T,U,V,X, and Y
0053 F880         LDI #80   ..respectively) are sampled in
0055 73           STXD      ..order up to the last channel.
0056 F802B6 START:#02->R6.1 ..The buffer
0059 94           GHI 4     ..starts at location
005A A6           PLO 6     ..#0200 (X=6).
005B 8A           GLO A     ..Use RF to keep track of
005C BF           PHI F     ..how much room is remaining
005D F8FFAF       #FF->RF.0 ..in the buffer.
0060 9A     CYCLE:GHI A     ..Start of a Channel scan.
0061 A2           PLO 2
0062 F80FA1       #0F->R1.0 ..Select the first MUX channel
0065 63           ,#63      ..before the scan starts.
0066 21           DEC R1
0067 3E67   WAIT1:BN3 WAIT1 ..Wait for a 0 to 5 V
0069 3669   WAIT2:B3 WAIT2  ..transition from the clock.
006B 63      NEXT:,#63      ..Select the MUX channel.
006C C4           ,#C4      ..Short Pause (NOP)
006D 63           ,#63      ..Commence an A/D conversion.
006E E6           SEX 6
006F 3F6F    BUSY:BN4 BUSY  ..Wait for end of conversion.
0071 6B           ,#6B      ..Input the data byte and store
0072 60           ,#60      ..in the next buffer location
0073 E1           SEX 1
0074 22           DEC 2     ..Is this the last channel?
0075 82           GLO 2     ..If no, then start the
0076 3A6B         BNZ NEXT  ..next conversion.
0078 2F           DEC F     ..If Yes, check to see if
0079 9F           GHI F     ..the buffer is full.
```

[illegible]

```
LOC  COSMAC CODE  SOURCE LINE
007A 3A60         BNZ CYCLE ..If no, continue the cycle.
007C F880B3       #80->R3.1 ..Else get ready for a
007F F891A3       #91->R3.0 ..Tape write.
0082 E4           SEX 4
0083 6300         ,#6300    ..Turn the cassette on.
0085 EC           SEX C
0086 F802B6       #02->R6.1 ..Tape write
0089 94           GHI 4     ..initializations.
008A A6           PLO 6
008B 8B           GLO B
008C AE           PLO E
008D D3           SEP 3     ..Call RCA Tape-write.
008E 61           ,#61      ..Turn Video off.
008F E4           SEX 4
0090 6380         ,#6380    ..Turn the cassette off.
0092 E1           SEX 1
0093 3056         BR START  ..Start over.
0095 000000000000 ,#000000000000 ..Filler
009B 0000000000   ,#0000000000
009B              ..-------------------------------------
009B              ..KEYPAD INPUT SUBROUTINE
009B
00A0 D4       RET:SEP 4     ..Return to Main.
00A1 F800B2       #00->R2.1 ..Set M(X) to
00A4 F8FFA2       #FF->R2.0 ..#00FF (a place to
00A7 E2           SEX 2     ..store value of key).
00A8 92           GHI 2     ..set D to zero
00A9 FC01   NEXT2:ADI #01   ..This loop  continuously
00AB FA0F         ANI #0F   ..scans the hexkeys until
00AD 52           STR 2     ..one of them is pushed.
00AE 62           ,#62
00AF 22           DEC 2
00B0 3E6B         BN3 NEXT
00B2 7B           ,#7B      ..Beep when a key is pressed.
00B3 F820BF       #20->RF.1
00B6 2F      CONT:DEC F     ..Debounce delay loop.
00B7 9F           GHI F
00B8 3AB6         BNZ CONT
00BA 36BA    HERE:B3 HERE   ..Turn off tone when the key
00BC 7A           ,#7A      ..is released.
00BD F0           LDX       ..Recall value of Key to D.
00BE 30A0         BR RET    ..Goto RET.
00C0              ..--------------------------------------
```

```
LOC  COSMAC CODE  SOURCE CODE
0000              ..---------------------------------------
0000              ..COSMAC-AMDAHL INTERFACE SOFTWARE
0000              ..WRITTEN BY PAUL HOPPS (1982)
0000              ..---------------------------------------
0000              ..REGISTER DESIGNATIONS:
0000              ..R0   Delay loop counter
0000              ..R1   End-of-line delay P counter
0000              ..R2   ASCII message pointer
0000              ..R2.0 Character per line counter
0000              ..R3   RCA Read-Tape-Routine P counter
0000              ..R4   Main program counter
0000              ..R5   Serial Output subroutine P counter
0000              ..R6   Buffer location pointer
0000              ..R7   Keyboard Input P counter
0000              ..R7.0 Parity Register
0000              ..R8.1 Number of channels of data
0000              ..R9.1 Total pages in buffer
0000              ..RA.0 ASCII ones digit
0000              ..RB.0 Temporary ASCII character storage
0000              ..RB.1 Number of characters per line
0000              ..RC   RCA Read-a-Bit subroutine
0000              ..RD   Delay routine program counter
0000              ..RE   Starting address of buffer
0000              ..RF   Workhorse register
0000              ..---------------------------------------
0000 7A           ,#7A        ..Initializations:
0001 90B1B4B5BABBBD R0.1->R1.1,R4.1,R5.1,RA.1,RB.1,RD.1
0008 F80CA4       #0C->R4.0 ..Make R4 the main program
000B D4           SEP 4     ..counter.
000C F801B7       #01->R7.1
000F F803A7       #03->R7.0
0012 D7           SEP 7     ..Input the number of channels
0013 B8           PHI 8     ..on the tape (store in R8.1).
0014 D7           SEP 7     ..Input the number of pages
0015 98AF         R8.1->RF.0 ..per channel on the tape.
0017 9A     PAGES:GHI A     ..Compute the total number of
0018 F4           ADD       ..pages per block by using
0019 BA           PHI A     ..multiple additions (store
001A 2F           DEC F     ..result in RA.1).
001B 8F           GLO F
001C 3A17         BNZ PAGES
001E D7           SEP 7     ..Input Scans/line to be printed.
001F 98AF         R8.1->RF.0
0021 9B     BYTES:GHI B     ..Compute the number of data
0022 F4           ADD       ..values to be stored per
0023 BB           PHI B     ..line of file in the Host
0024 2F           DEC F     ..computer (store in RB.1).
0025 8F           GLO F
0026 3A21         BNZ BYTES ..Initializations:
0028 F8ADA1       #AD->R1.0 ..Delay address (#00AD)
002B F8BDA5       #BD->R5.0 ..P-to-S address (#00BD)
002E F827A2       #27->R2.0 ..Message address (#0127)
0031 F881BC       #81->RC.1 ..need by RCA routines
```

```
LOC  COSMAC CODE  SOURCE LINE
0034 F8F4AD       #F4->RD.0 ..WaitC address (#00F4)
0037 F819AF       #19->RF.0 ..25 characters in message
003A 3E3A   HERE: BN3 HERE  ..Wait for repeat of last key
003C 42     MSG1: LDA 2     ..before starting to transmit.
003D D5           SEP 5     ..Send the message
003E 2F           DEC F     ..'$COPY *MSOURCE* -DATA'
003F 8F           GLO F     ..to the Host computer
0040 3A3C         BNZ MSG1
0042 F80D   LOAD: LDI #0D   ..Send a carriage return.
0044 D5           SEP 5
0045 F80A         LDI #0A   ..Send a linefeed.
0047 D5           SEP 5
0048 D1           SEP 1     ..Delay for system response.
0049 E4           SEX 4
004A 6300         ,#6300    ..Turn the recorder on.
004C EC           SEX C
004D F802B6       #02->R6.1 ..Initializations for the
0050 94           GHI 4     ..RCA tape read (the buffer
0051 A6           PLO 6     ..starts at #0200).
0052 9AAEBF       RA.1->RE.0,RF.1
0055 F880B3       #80->R3.1
0058 F8C2A3       #C2->R3.0
005B D3           SEP 3     ..Execute RCA tape read.
005C 61           ,#61      ..Turn the video off.
005D E4           SEX 4
005E 6380         ,#63,#80  ..Turn the recorder off.
0060 F802B6       #02->R6.1
0063 94           GHI 4     ..Prepare to send the data.
0064 A6           PLO 6
0065 E6           SEX 6
0066 F8FFAF       #FF->RF.0 ..RF counts down chars/block.
0069 F830         LDI #30   ..Send a '0' a block separater.
006B D5           SEP 5
006C 9BA2   NEXTL:RB.1->R2.0 ..R2 counts down chars/line.
006E F80D         LDI #0D   ..Send a carriage return.
0070 D5           SEP 5
0071 F80A         LDI #0D   ..Send a linefeed.
0073 D5           SEP 5
0074 D1           SEP 1     ..Delay for system response.
0075 F820   NEXTC:LDI #20
0077 D5           SEP 5     ..Send a 'space'.
0078 72           ,#72      ..Get the next data byte.
0079 AA           PLO A
007A F82FAB       #2F->RB.0 ..Convert hundreds digit to
007D 8A           GLO A     ..ASCII by repeatedly
007E AA     LOOPH:PLO A     ..subtracting #64=100 until
007F FF64         SMI #64   ..the remainder is less than
0081 1B           INC B     ..100. Add #30 to the hex
0082 337E         BDF LOOPH ..result.
0084 8B           GLO B
0085 D5           SEP 5     ..Transmit hundreds digit.
0086 F82FAB       #2F->RB.0 ..Convert tens digit to
0089 8A           GLO A     ..ASCII by repeatedly
```

```
LOC  COSMAC CODE  SOURCE LINE
008A AA     LOOPT:PLO A      ..subtracting #0A=10
008B FF0A         SMI #0A    ..from the remainder to the
008D 1B           INC B      ..hundreds loop.
008E 338A         BDF LOOPT
0090 8B           GLO B
0091 D5           SEP 5      ..Send tens digit.
0092 8A           GLO A      ..Ones digit is remainder
0093 FC30         ADI #30    ..of tens loop.
0095 D5           SEP 5      ..Send the ones digit.
0096 22           DEC 2
0097 2F           DEC F
0098 9F           GHI F      ..If this is the last number
0099 3242         BZ LOAD    ..in the buffer goto LOAD.
009B 82           GLO 2      ..Is this the last character
009C 3A75         BNZ NEXTC ..in the line? If no, goto NEXT.
009E 306C         BR NEXTL   ..If Yes, goto NEXTL (nextline)
00A0 000000000000 ,#000000000000 ..Filler
00A6 000000000000 ,#000000000000
00AC              ..----------------------------------------
00AC              ..END-OF-LINE DELAY SUBROUTINE
00AC
00AC D4      RETD:SEP 4      ..Return to Main.
00AD F820B0 DELAY:#20->R0.1 ..Delay for AMDAHL response.
00B0 20     LOOPD:DEC 0
00B1 90           GHI 0
00B2 3AB0         BNZ LOOPD
00B4 30AC         BR RETD    ..Goto RETD.
00B6 00000000000  ,#000000000000
00BC              ..----------------------------------------
00BC              ..PARALLEL TO SERIAL ROUTINE
00BC
00BC D4      RETS:SEP 4      ..Return to Main.
00BD 7A      PTOS:,#7A       ..Turn Q tone off.
00BE A8           PLO 8      ..Binary data value-->R8.0
00BF F800A7       #00->R7.0 ..Initialize parity reg (odd)
00C2 F807A9       #07->R9.0 ..and bit counter (7).
00C5 7B           ,#7B       ..Turn Q tone on.
00C6 F83B         LDI #3B
00C8 DD           SEP D      ..Send start bit and then
00C9 88     SHFTR:GLO 8      ..start shifting out data bits.
00CA F6           SHR
00CB A8           PLO 8      ..R8.0 Holds remaining bits.
00CC 33D1         BDF HIGH   ..Turn Q on for a low bit.
00CE 7B           ,#7B
00CF 30D3         BR NEXTB   ..Get next bit.
00D1 7A      HIGH:,#7A       ..Turn Q off for a high bit.
00D2 17           INC 7      ..Increment partiy register.
00D3 F83A   NEXTB:LDI #3A
00D5 DD           SEP D      ..delay for data bit(600 BAUD).
00D6 29           DEC 9      ..Is this the last data bit?
00D7 89           GLO 9      ..If no, get the next bit
00D8 3AC9         BNZ SHFTR ..else send out the parity bit.
00DA 87           GLO 7
```

```
LOC  COSMAC CODE  SOURCE LINE
00DB F6           SHR
00DC A7           PLO 7
00DD 33E2         BDF HPAR
00DF 7B           ,#7B
00E0 30E5         BR STOPB
00E2 7A     HPAR: ,#7A
00E3 30E5         BR STOPB
00E5 F81B   STOPB:LDI #1B
00E7 DD           SEP D      ..Delay for parity bit.
00E8 7A           ,#7A
00E9 F8FF         LDI #40    ..send out stop bit.
00EB DD           SEP D
00EC 30BC         BR RETS    ..Return.
00EE 0000000000   ,#0000000000
00F3              ..-------------------------------------
00F3              ..SERIAL OUTPUT BIT DELAY
00F3
00F3 D5      RETC:SEP 5      ..Return to P-TO-S routine
00F4 A0     WAITC:PLO 0      ..Single bit delay loop.
00F5 20     LOOPC:DEC 0
00F6 80           GLO 0
00F7 3AF5         BNZ LOOPC
00F9 30F3         BR RETC    ..Goto retc.
00FB 00000000000000  ,#00000000000000
0102              ..-------------------------------------
0102              ..KEYBOARD INPUT ROUTINE
0102
0102 D4      RETK:SEP 4      ..Return to Main.
0103 F801B2 INPUT:#01->R2.1 ..Set M(X) to #01FF (a place
0106 F8FFA2       #FF->R2.0 ..to store the value of key).
0109 E2           SEX 2
010A 92           GHI 2
010B FC01   NEXTK:ADI #01    ..This loop continuously
010D FA0F         ANI #0F    ..scans the hex keys until one
010F 52           STR 2      ..of them is pressed.
0110 62           ,#62
0111 22           DEC 2
0112 3E0B         BN3 NEXTK
0114 7B           ,#7B       ..Beep when a key is pressed.
0115 F820BF       #20->RF.1
0118 2F     LOOPK:DEC F      ..Debounce delay loop.
0119 9F           GHI F
011A 3A18         BNZ LOOPK
011C 361C    PKEY:B3 PKEY    ..Turn off tone when the
011E 7A           ,#7A       ..key is released.
011F F0           LDX        ..Recall value of key to D
0120 3002         BR RETK    ..and go to RETK.
0122 0000000000   ,#0000000000
0127              ..-------------------------------------
0127              ..ASCII MESSAGES FOR HOST
0127              ..
0127 24           ,#24  ..$
0128 43           ,#43  ..C
```

```
LOC  COSMAC CODE  SOURCE CODE
0129 4F           ,#4F  ..O
012A 50           ,#50  ..P
012B 59           ,#59  ..Y
012C 20           ,#20  ..SPACE
012D 2A           ,#2A  ..*
012E 4D           ,#4D  ..M
012F 53           ,#53  ..S
0130 4F           ,#4F  ..O
0131 55           ,#55  ..U
0132 52           ,#52  ..R
0133 43           ,#43  ..C
0134 45           ,#45  ..E
0135 2A           ,#2A  ..*
0136 20           ,#20  ..SPACE
0137 2D           ,#2D  ..-
0138 44           ,#44  ..D
0139 41           ,#41  ..A
013A 54           ,#54  ..T
013B 41           ,#41  ..A
013C 28           ,#28  ..(
013D 2A           ,#2A  ..*
013E 4C           ,#4C  ..L
013F 29           ,#29  ..)
0140              ..--------------------------------------
```

```
LOC  COSMAC CODE  SOURCE CODE
8091              ..-------------------------------------
8091              ..WRITE TO TAPE ROUTINE
8091              ..FROM RCA VIP-711 OPERATING SYSTEM.
8091              ..-------------------------------------
8091              ..REGISTER DESIGNATIONS:
8091              ..R3   Program counter
8091              ..R6   Pointer to next data byte
8091              ..R7   Parity register
8091              ..R7.1 Holds byte being written
8091              ..R9   Sync tone timer
8091              ..R9.0 Bit counter
8091              ..RC   P counter to Bit-write subroutine
8091              ..RE   Page counter (last byte makes it 0)
8091              ..-------------------------------------
8091 F86FAC WRITE:#6F->RC.0 ..Initializations, RC=#816F
8094 F840B9       #40->R9.1
8097 93      SYNC:GHI 3     ..Start Sync tone, send out
8098 F6           SHR       ..a high bit.
8099 DC           SEP C
809A 29           DEC 9     ..Decrement sync counter and
809B 99           GHI 9     ..continue sending high bits
809C 3A06         BNZ SYNC  ..for 5 seconds.
809E F810A7 NEWBY:#10->R7.0 ..Reset parity register (Odd).
80A1 F808A9       #08->R9.0
80A3 46           LDA R6    ..Get next data byte to send
80A4 B7           PHI 7
80A5 93           GHI 3     ..Send out a start bit
80A6 FE           SHL
80A8 DC           SEP C
80A9 86           GLO 6     ..Is this the end of the page?
80AA 3AAD         BNZ NXTBT ..If yes, then decrement the
80AC 2E           DEC E     ..page counter.
80AD 97     NXTBT:GHI 7     ..Get remaining data byte
80AE F6           SHR       ..and write next bit to tape.
80AF B7           PHI 7
80B0 DC           SEP C
80B1 29           DEC 9     ..Is this the last bit?
80B2 89           GLO 9
80B3 3A1D         BNZ NXTBT ..If no, get next bit.
80B5 17           INC 7     ..If yes, finish up the
80B6 87           GLO 7     ..parity calculation and write
80B7 F6           SHR       ..it to tape.
80B8 DC           SEP C     ..
80B9 8E           GLO E     ..Is this the last byte?
80BA 3A9E         BNZ NEWBY ..If no, then get new byte,
80BC DC           SEP C     ..else send a stop bit.
80BD 69     VIDEO:,#69      ..Turn video display on.
80BE 26           DEC 6     ..Point at last byte written.
80BF D4           SEP 4     ..Call video interrupt routine.
80C0 3030    END1:BR END1   ..Terminate in this loop.
80C2              ..-------------------------------------
```

```
LOC  COSMAC CODE  SOURCE LINE
                  ..-----------------------------------------
80C2              ..TAPE READ ROUTINE
80C2              ..FROM RCA VIP-711 OPERATING SYSTEM.
80C2              ..-----------------------------------------
80C2              ..REGISTER DESIGNATIONS:
80C2              ..R6   Pointer to data buffer
80C2              ..R7.0 Partiy register
80C2              ..R7.1 Temporarily stores incoming byte
80C2              ..R9   Sync tone timer
80C2              ..R9.0 Bit counter
80C2              ..RC   Read-a-bit P Counter
80C2              ..RE.0 Page counter
80C2              ..-----------------------------------------
80C2 F883AC  READ:#83->RC.0 ..Initialize, RC=#8183
80C5 F80AB9 AGAIN:#0A->R9.1
80C8 DC      TONE:SEP C     ..Read in a bit
80C9 33C5         BDF AGAIN ..Keep trying until a 1-bit.
80CB 29           DEC 9     ..Are there 2560 high bits
80CC 99           GHI 9     ..in a row?
80CD 3AC8         BNZ TONE
80CF DC    WAITST:SEP C     ..Yes, so it must be a sync;
80D0 3BCF         BNF WAITST..Just wait for a start bit.
80D2 F809A9A7     #09->R9.0,R7.0
80D6 97      NEXT:GHI 7     ..Get data byte so far and
80D7 76           SHRC      ..shift on the next bit.
80D8 B7           PHI 7
80D9 29           DEC 9
80DA DC           SEP C     ..Read in next bit.
80DB 89           GLO 9     ..Is this the last bit?
80DC 3AD6         BNZ NEXT  ..No, then get the next one.
80DE 87           GLO 7     ..Yes, so check the parity.
80DF F6           SHR
80E0 33E3         BDF PARTY ..Turn Q on if there is
80E2 7B           ,#7B      ..a parity error.
80E3 97     PARTY:GHI 7     ..Store finished byte in
80E4 56           STR 6     ..the next buffer location
80E5 16           INC 6
80E6 86           GLO 6     ..Is this end of a page?
80E7 3ACF         BNZ WAITST ..No, then read next byte.
80E9 2E           DEC E     ..Yes, decrement page counter.
80EA 8E           GLO E     ..Is this the very last byte?
80EB 3ACF         BNZ WAITST ..No, then get next else
80ED 30BD         BR #VIDEO ..goto VIDEO and terminate.
80EF              ..-----------------------------------------
```

```
LOC  COSMAC CODE  SOURCE LINE
816E                ..----------------------------------------
816E                ..SUBROUTINE TO WRITE A BIT TO TAPE.
816E                ..FROM RCA VIP-711 OPERATING SYSTEM.
816E                ..----------------------------------------
816E                ..REGISTER DESIGNATIONS:
816E                ..R3  P counter of calling program
816E                ..R7  Parity register
816E                ..RC  P counter of this subroutine
816E                ..RF  timing counter
816E                ..----------------------------------------
816E D3        RET1:SEP 3     ..Return to Tape-write.
816F F80A           LDI #0A   ..Initialize timer to 10.
8171 3B76           BNF ZERO  ..Is this a zero bit?
8173 F820           LDI #20   ..If yes, then let timer be 32
8175 17             INC 7     ..and increment parity reg.
8176 7B        ZERO:,#7B      ..Turn Q on, start of bit.
8177 BF             PHI F
8178 FF01     MINUS:SMI #01   ..Count down timer with
817A 3A78           BNZ MINUS..Q turned on. When timer
817C 396E           BNQ RET1  ..is zero, turn Q off and
817E 7A             ,#7A      ..repeat timer loop with Q
817F 9F             GHI F     ..OFF. When finished
8180 3078           BR MINUS  ..return to calling routine.
8182                ..----------------------------------------
8182                ..SUBROUTINE TO READ IN A BIT FROM TAPE
8182                ..FROM RCA VIP-711 OPERATING SYSTEM
8182                ..----------------------------------------
8182                ..REGISTER DESIGNATIONS:
8182                ..R3   P counter of calling program
8182                ..R7   Parity register
8182                ..RC   P counter of this subroutine
8182                ..----------------------------------------
8182 D3        RET2:SEP 3     ..Return to Tape-Read.
8183 F810           LDI #10   ..Initialize Timer to 10.
8185 3D85      WAIT:BN2 WAIT  ..Wait for start of bit.
8187 3D8F     TIMER:BN2 OUT   ..If the loop is exited from
8189                          ..this point then it must
8189                          ..be a low bit.
8189 FF01           SMI #01   ..Continue timing.
818B 3A87           BNZ TIMER..If we fall through the
818D 17             INC 7     ..timing loop it must be
818E                          ..a high bit. Increment the
818E                          ..parity register.
818E 9C             GHI C     ..Get current byte.
818F FE         OUT:,#FE      ..(SHL)Shift out this bit.
8190 3590     WAIT2:B2 WAIT2  ..Wait for bit transition
8192 3082           BR RET2   ..if necessary then return.
8194                ..----------------------------------------
```